TECHNISCHE UNIVERSITÄT MÜNCHEN

Fakultät für Informatik
Computer Aided Medical Procedures & Augmented Reality / I16

Signed Distance Fields for Rigid and Deformable 3D Reconstruction

Miroslava Slavcheva

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr. Daniel Cremers
Prüfer der Dissertation:		
	1.	Priv.-Doz. Dr. Slobodan Ilic
	2.	Hon.-Prof. Dr. Michael Black

Die Dissertation wurde am 14.06.2018 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 23.09.2018 angenommen.

Abstract

Capturing three-dimensional environments is a key task in the growing fields of virtual and augmented reality. Methods have to work at sensor frame rates and be extremely accurate to ensure a credible reconstruction. Moreover, they have to be able to not only recover the geometry of a fixed scene, but also capture it when people are moving and interacting with objects in it. This thesis addresses the task of 3D reconstruction of both static and dynamic objects and scenes scanned with a single hand-held RGB-D camera, without any markers or prior knowledge.

Reconstructing rigid environments requires estimating the six degrees-of-freedom camera pose at every time instance, and subsequently fusing the acquired data into a geometrically consistent computer model. The task of reconstructing deformable objects is more challenging, as additionally the non-rigid motion that occurred in every frame has to be determined and factored out. Solutions are typically based on variants of the iterative closest points (ICP) algorithm, which iteratively establishes correspondences and minimizes the distance between two point sets. While this approach is general and versatile, it is dependent on a good initialization and a low amount of noise.

Recently, point-to-implicit approaches have shown higher robustness than ICP for rigid registration. They align a point cloud with the zero level set of a signed distance field (SDF). The SDF is an implicit surface representation, stored as a voxel grid in which outside areas have positive values and inside areas have negative ones, leaving the surface as the zero-valued interface. It permits registration to be done as a direct minimization without correspondence search.

Inspired by this, we propose to tackle both the rigid and deformable reconstruction problems via implicit-to-implicit alignment of SDF pairs. In the static case, we obtain more accurate pose estimates with a framework that permits straightforward incorporation of various additional constraints, such as surface colour and orientation. We start with the reconstruction of small- to medium-scale household objects and demonstrate how to extend the approach to larger spaces such as rooms. To this end, we develop a limited-extent volume strategy that restricts registration to the most geometrically distinctive regions of a scene, leading to significantly improved rotational motion estimation.

Finally, we adapt our approach to dynamic scenes by modifying our implicit-to-implicit approach so that new data is incremented appropriately. For this purpose we evolve an initial SDF to a target SDF by imposing rigidity constraints that require the underlying deformation field to be approximately Killing, *i.e.* volume-preserving and generating locally isometric motions. Alternatively, we employ gradient flow in the smooth Sobolev space, which favours global deformations over finer-scale details. These strategies also circumvent explicit correspondence search and thus avoid the repeated conversion between SDF and mesh representations that other techniques entail. Nevertheless, we ensure that correspondence information can be recovered by proposing two strategies based on Laplacian eigenfunctions, which are known to encode natural deformation patterns. Thanks to the used SDF representation, our non-rigid reconstruction approach is able to handle topological changes and fast motion, which are major obstacles for existing approaches.

Zusammenfassung

Die Erfassung dreidimensionaler Umgebungen ist eine zentrale Aufgabe in den wachsenden Feldern der virtuellen und erweiterten Realität. Methoden hierfür müssen echtzeitig und extrem genau sein, um eine glaubwürdige Rekonstruktion zu gewährleisten. Darüber hinaus müssen sie in der Lage sein, nicht nur die Geometrie einer statischen Szene zu erfassen, sondern diese auch wiederherzustellen, wenn sich Personen bewegen und mit Objekten interagieren. Diese Dissertation befasst sich mit der 3D-Rekonstruktion von statischen und dynamischen Objekten und Szenen, die mit einer handgeführten RGB-D Kamera ohne Marker oder Vorkenntnisse aufgenommen wurden.

Das Rekonstruieren starrer Umgebungen erfordert zu jedem Zeitpunkt die Schätzung von sechs Freiheitsgraden der Kamerapose und anschließendes Kombinieren der erfassten Daten zu einem geometrisch konsistenten Computermodell. Für verformbare Objekte ist dies schwieriger, da zusätzlich der nicht-starre Bewegungsanteil herausgerechnet werden muss. Lösungen basieren typischerweise auf Varianten des Iterative Closest Points (ICP) Algorithmus, der die Korrespondenzen zwischen zwei Punktwolken festlegt. Obwohl dieser Ansatz allgemein einsetzbar ist, braucht er für ein gutes Rekonstruktionsergebnis eine gute Initialisierung sowie möglichst wenig Rauschen.

Kürzlich haben Punkt-zu-implizite Ansätze eine höhere Robustheit als ICP für starre Registrierung gezeigt. Sie richten eine Punktwolke mit dem Nullpegel eines vorzeichenbehafteten Distanzfeldes (Signed Distance Field - SDF) aus. Das SDF ist eine implizite, als Voxelgitter gespeicherte Oberflächendarstellung, in der äußere Bereiche positive und innere Bereiche negative Werte haben, wobei die Oberfläche als nullwertige Schnittstelle verbleibt. Es ermöglicht eine Registrierung als direkte Minimierung ohne Korrespondenzsuche.

Davon ausgehend schlagen wir vor, sowohl starre als auch deformierbare Rekonstruktionsprobleme durch implizite-zu-implizite Ausrichtung von SDF-Paaren zu lösen. Im statischen Fall erhalten wir ein Framework zur genaueren Posenschätzung, welches die direkte Integration verschiedener Randbedingungen, wie z. B. Oberflächenfarbe und Orientierung, erlaubt. Wir beginnen mit der Rekonstruktion von kleinen bis mittelgroßen Haushaltsobjekten und zeigen, wie der Ansatz auf größere Räume erweitert werden kann. Zu diesem Zweck entwickeln wir eine Strategie mit Volumen begrenzter Ausdehung, welche die Registrieurng auf die geometrisch am stärksten ausgeprägten Bereiche einer Szene einschränkt, was zu einer signifikant verbesserten Rotationsbewegungsschätzung führt.

Schließlich erweitern wir unseren Ansatz auf dynamische Szenen. Wir beginnen mit demselben impliziten zu impliziten Ansatz, passen ihn jedoch so an, dass neue Daten inkrementell angemessen hinzugefügt werden. Zu diesem Zweck entwickeln wir ein initiales SDF zu einem finalen SDF. Dafür verwenden wir Steifigkeitseinschränkungen, die erfordern, dass das zugrundeliegende Deformationsfeld annähernd Killing ist, d. h. lokal isometrische Bewegungen erzeugt und daher Volumen erhält. Alternativ verwenden wir einen Gradientenfluss in dem glatten Sobolev-Raum, welcher statt feineren Details globale Deformationen begünstigt. Diese Strategien umgehen auch die explizite Korrespondenzsuche und vermeiden somit die wiederholte Konvertierung zwischen SDF- und triangulierten Netzdarstellungen, die andere Techniken mit sich bringen. Nichtsdestotrotz stellen wir durch zwei Strategien basierend auf Laplace-Eigenfunktionen, welche natürliche Deformationsmuster codieren, sicher, dass Korrespondenzinformation wiederhergestellt werden kann. Dank der verwendeten SDF-Darstellung ist unser Ansatz in der Lage, mit topologischen Veränderungen und schnellen Bewegungen umzugehen, welche große Hindernisse für bestehende Ansätze darstellen.

Acknowledgments

Having just written the last words of my dissertation, I realize how lucky I am to still love the subject of my work after all the ups and downs along this journey. This would not have been possible without the consistent support and guidance of my supervisor PD Dr. Slobodan Ilic, to whom I am eternally grateful. It would have also not been possible without the wise words and fantastic collaboration with my advisor, Dr. Maximilian Baust. I would also like to thank Prof. Dr. Nassir Navab and Dr. Claudio Laloni for providing access to their facilities and research groups at the Technical University of Munich and Siemens AG respectively. Further, I thank Prof. Dr. Michael Black for serving on my PhD committee.

I had the chance to meet and work with many incredible researchers during my PhD. First and foremost, I want to mention Wadim Kehl and Fausto Milletari who consistently inspired me with their light-hearted approach to research and life. I also want to thank my office mate, Tolga Birdal, for the refreshing conversations and shared suffering. I am further thankful to Prof. Dr. Daniel Cremers for the collaboration and for exposing me to the work at his Computer Vision Group. I would also like to express gratitude to the other colleagues from the Chair for Computer Aided Medical Procedures at TUM and the Department for Digital Perception at Siemens: Haowen Deng, Sergey Zakharov, Carmel Kozlov, Federico Tombari, Christian Rupprecht, Chun-Hao Paul Huang, Oliver Scheel, Keisuke Tateno, David Tan, Vasilis Belagiannis, Fabian Manhardt, Johanna Wald, Salvatore Virga, Benjamin Busam, Iro Laina, Sailesh Conjeti, Frank Forster, Patrick Wissmann, Helmuth Euler, and numerous others. I would especially like to point out the people who agreed to endure the pleasure of being models for the sequences in my papers, namely Alexander Seeber, Adrian Haarbach and Denys Korzh. Furthermore, I am very happy to have met and discussed research with numerous people at conferences or over email, and often also become friends with them.

Last but definitely not least, I thank my parents, Detelinka and Georgi, for their constant support and encouragement throughout all my studies. Likewise, I thank my friends, and especially Milena, Mahyar and Kiril, for always being there to listen about paper submissions, reviews and deadline stress.

Lastly, I want to thank the anonymous reviewers who rejected my first ever paper submission. Twice. Although this experience was rather painful, it taught me to express myself clearly and aim for challenging problems, proving that any part of a PhD is a useful lesson. Now that several submission cycles have passed smoothly, I am excited to discover what new challenges the future will bring.

– Mira

Contents

List of Figures

List of Tables

Part I

Introduction and Background

1 Introduction

Human beings have an innate interest in their environment. Seeking to find where we came from and what our purpose is, we have been studying our surrounding world through many disciplines going from physics and chemistry to philosophy. Among these, *computer vision* is the one that enables us to capture our habitats using various imaging modalities and subsequently reason about them.

Our world has three spatial dimensions, which makes the task of *3D reconstruction* exceptionally important and exciting. It deals with the creation of three-dimensional digital models of the surfaces of objects in the real world from sensor data. They can be utilized for various inspection, planning and navigation applications.

Typically the process starts with the acquisition of images from multiple viewpoints. They are then *registered*, *i.e.* brought into a common reference frame via rigid camera pose estimation or non-rigid alignment [27] for instance. Finally, the data is *fused* into a geometrically consistent reconstruction [13].

If only a monocular sensor is available, a 3D model is obtained via methods such as structure from motion [49, 84], shape from stereo [77, 189], shape from silhouettes [124] or shape from shading [90, 244]. However, this modality entails an inherent scale ambiguity [84], which is resolved by 3D scanners. The recent advancements in 3D sensing technology have made real-time dense surface reconstruction achievable. In particular, the rise of inexpensive consumer-grade depth cameras, such as the Microsoft Kinect and PrimeSense Carmine [142], have lead to the development of a variety of compelling techniques.

While the majority of these methods focus on reconstructing static environments, dynamic scene capture has recently been attracting more and more research efforts. Modelling rigid objects requires estimating the position and orientation of the camera at every instant, *i.e.* its 6 degrees-of-freedom (DoF) pose [37]. In contrast, non-rigid reconstruction is a problem of much higher dimensionality, as each point might have followed a different trajectory from the rest, yielding infinitely many possible solutions [74]. This thesis is aimed at recovering the geometry of both rigid and deformable surfaces using the input of a single RGB-D sensor.

1.1 Motivation

Real-time camera tracking and dense surface capture are vital components of augmented and virtual reality systems. While research in these fields has a long history, it is still subject to a number of open questions.

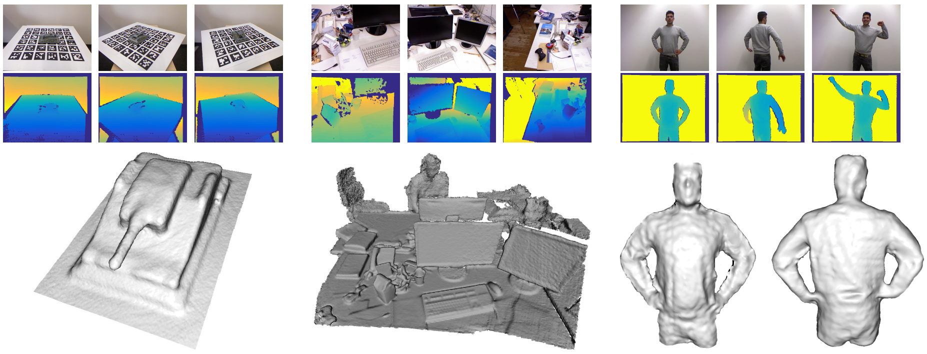
The earliest techniques for structure from motion (SfM) [70] and multi-view stereo (MVS) [189] were capable of producing a sparse 3D reconstruction from images acquired with a monocular RGB sensor. Later on, research on simultaneous localization and mapping (SLAM) lead to the development of real-time systems such as Parallel Tracking and Mapping (PTAM) [117] and MonoSLAM [48]. Subsequently, also methods that recover dense 3D models by replacing feature tracking with whole image alignment emerged [62, 158, 207]. However, these approaches rely on suitable scene illumination and yield reconstructions of unknown scale [84].

3D scanners do not suffer from these issues as they capture a point cloud or depth map representation of the observed scene from a given vantage point. Range-finders have been successfully employed for the accurate reconstruction of small and large objects, for example in the Digital Michelangelo Project [127]. While such laser scanners make very precise measurements, the acquisition process is too slow to permit real-time applications. The introduction of high-frame rate time-of-flight (ToF) and structured light cameras that capture both colour and depth (RGB-D) opened up the possibility for dense surface modelling at 30 Hz.

The first approaches that utilized RGB-D sensors were still based on sparse feature matching and only leveraged the available per-pixel depth information for the subsequent iterative closest point (ICP) [15, 37] registration [86]. Shortly after, the seminal KinectFusion work came about [157]. It was the first system for creating dense volumetric reconstructions of static scenes with hand-held Kinect sensors in real time using the full depth images. The approach stores the recovered geometry in a continuously growing model, and tracks the 6 DoF camera motion against it via point-to-plane ICP in a frame-to-model fashion. It then inspired numerous extensions coping with larger volumes [161, 240] or correcting errors via surface re-integration [46].

A key component of these systems is the volumetric fusion of Curless and Levoy [45]. The geometry is stored in a 3D grid representing a truncated signed distance field (SDF), where each voxel contains the distance to the nearest surface. It is a type of implicit structure in which the inside of objects has negative values, the outside is positive, and the actual surface is at the zero-level set interface [163]. This not only provides a convenient way to extract a mesh via marching cubes [136] or ray tracing [6, 166], but also allows for continuous surface refinement through the averaging of multiple measurements.

Identifying the advantageous properties of the SDF representation, point-to-implicit follow-ups of KinectFusion replaced the used ICP registration with a direct alignment of an incoming point cloud with the zero-level set of the cumulative SDF [30, 35]. This strategy circumvents correspondence estimation, thus gaining both speed and accuracy. Inspired by this, we propose to study the direct alignment of pairs of signed distance fields in this thesis.



(a) Static objects.

(b) Static scenes.

(c) Non-rigid surfaces.

Figure 1.1: **Types of 3D reconstruction from a single RGB-D stream** addressed in this thesis. We target (a) small- and (b) large-scale rigid objects, as well as (c) non-rigidly moving surfaces, including topological changes.

Despite the major progress in mapping static environments, the reality remains that our world is dynamic, as people move and interact with objects and with each other. Reconstructing non-rigid surfaces using a single RGB-D device is extremely challenging due to the inherent ambiguity of the problem. DynamicFusion was the first method to modify a KinectFusion-like system into one that can simultaneously track and reconstruct a moving surface [156]. To this end, it estimates a warp field that deforms the global model to best explain the currently observed depth measurements. Several follow-ups further improved it by integrating SIFT features [138] to anchor the model and handle tangential motions [98] or by adding surface albedo constraints to increase robustness and allow for less contrived movements [81]. Nevertheless, these approaches cannot cope with changing topology due to the underlying mesh-based correspondence estimation.

On the other hand, the SDF representation that we explore here entails no special treatment when topological changes occur [163, 243]. Therefore, we set out to apply it not only for the reconstruction of rigid objects and scenes, but also for non-rigid ones, as exemplified in Figure 1.1.

1.2 Objectives

Given the applications outlined so far, our first task is to develop a method for very accurate 6 DoF pose estimation that can be used for tracking the camera in all following methods. As trajectory estimation inevitably suffers from drift [122], the approach has to be easily extendible to pose refinement too. Driven by the intuition that a pair of SDFs will both steer towards optimal overlap without the need to determine explicit correspondence, we aim to accomplish our goal through direct SDF alignment. Naturally, the reconstructed models will also be represented via SDFs, and converted to mesh whenever needed for display.

We will investigate the applicability of such registration to both object and scene scanning. The difference between them is that typically objects are captured with an outside-in inward-facing trajectory, while larger spaces are explored with a SLAM-like inside-out one. It is known that different algorithms are usually better at one of these types [36], so we will examine the limitations of our approach here.

In the second part of the dissertation we will shift focus to reconstructing non-rigid environments. Our objective will be to find ways to combine the images from a single depth video into a 3D model by factoring out the non-rigid movement in every frame via SDF evolution. We will explore what additional constraints over the SDF or warp field are required to accomplish this goal, while being able to capture both rapid motion and topological changes. Finally, as SDF evolution typically loses track of correspondences [169, 256], we aim to recover them in our applications, as they are needed for tasks such as texture transfer, 4D video compression and character animation [44].

To sum up, our objectives are the following:

- precise 6 DoF tracking and refinement;
- capture of various scale static environments, going from household objects to large office spaces, with the same methodology in order to investigate its applicability to different settings;
- non-rigid reconstruction that works under topological changes and fast motion, and recovers correspondences.

1.3 Contributions

To fulfil the objectives listed in the previous section, we develop several novel algorithms that have the following contributions:

- **Correspondence-free alignment energy between pairs of signed distance fields.** Depending on the application, it is tuned for rigid or non-rigid reconstruction by only changing what the SDF generation depends on: a 6 DoF camera pose for the former, and a dense warp field for the latter.
- **Highly accurate 6 DoF camera tracking and pose optimization for rigid object and scene reconstruction** without explicit correspondence estimation, when the energy is dependent on the pose from which an SDF is generated. Furthermore, we propose various additional SDF-based constraints that make the energy more precise, like surface orientation constraints, or help it work on scenes with poorer geometry, such as photoconsistency constraints between voxel grids.
- **Reconstruction of non-rigidly moving surfaces, which is able to cope with changing topology and large motions** without explicit correspondence search, when the energy is dependent on the deformation field applied to one of the SDFs. We integrate different regularizers into a variational level set method variant and investigate two alternatives to

ensure that the warp field is geometrically plausible. On the one hand, we enforce it to be an approximately Killing vector field (AKVF) [200], so that it generates locally nearly isometric motions, achieving an effect similar to an as-rigid-as-possible regularizer over a mesh representation [201]. On the other hand, instead of adhering to the commonly used gradient defined via an L^2 inner product, we redefine it in the smooth Sobolev space H^1, resulting in gradient flow which leads to a favourable coarse-to-fine evolution behaviour and is less susceptible to local minima.

- **Voxel correspondence recovery in correspondence-free SDF evolution.** For moderate motions, we implicitly obtain correspondences via an additional data term which aligns the lowest-energy Laplacian eigenfunctions of the two shapes of interest, which we term *eigencolourings*, driven by the fact that they encode the deformation patterns that a shape can undergo. For larger motions, we explicitly match voxels by matching the signatures of the K lowest-frequency eigenfunctions in an approach that handles partial shapes via carefully designed outlier rejection.

- **Datasets for quantitative evaluation.** To be able to truly judge the performance of our methods, we strongly rely on numerical assessment of their accuracy. Whenever possible, we use existing datasets, but when we identify that they are not sufficient for thorough evaluation, we create our own ones, shown in Figure 1.2, and make them available to the public.
  In the case of rigid reconstruction, we have created a *3D-Printed RGB-D Object Dataset* [198], which contains five objects 3D-printed from original CAD models and scanned with a precise industrial sensor, a Kinect v1, and also rendered synthetically. While we consider our biggest contribution to be the provision of ground-truth 3D models, we also release the scanning trajectories estimated from a marker board. This permits comprehensive evaluation of both tracking and reconstruction fidelity on data with different quality and noise characteristics.
  For non-rigid reconstruction, we address the lack of ground-truth canonical models for evaluation of single-stream modelling in the *Deformable 3D Reconstruction Dataset* [193]. We use a couple of mechanical toys that have a rest pose in which we reconstruct them using a markerboard for pose estimation, before we record sequences in which they move non-rigidly. Although this does not permit every-frame evaluation, it is a first step to quantitative evaluation for non-rigid fusion from a single RGB-D camera.

1.4 Outline

This section provides a brief overview of each of the subsequent chapters. Most of the methods and material of this thesis are published or are under submission for a major conference or journal. Therefore, we additionally provide the work related to each chapter, and encourage the interested reader to consult the online material for video demonstrations of presented methods.

Chapter 2 We first provide the theoretical foundation upon which this thesis is built. In particular, we outline the RGB-D imaging process and projective geometry. In addition, we review registration methods for both rigid and non-rigid motion, including 6 DoF transformation parameterizations and deformation field regularization.

Chapter 3 This chapter presents the building block of all methods developed in this dissertation, the signed distance field (SDF). After an overview of its geometric properties, we outline the generation process we use and devise the energy for direct alignment of pairs of SDFs, dubbed *SDF-2-SDF energy*.

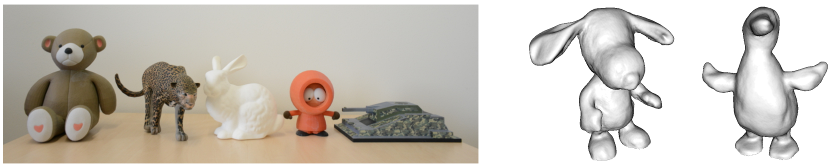
Chapter 4 Here we build upon the concepts developed in Chapter 3 and employ them for the task of highly accurate 3D reconstruction of small-scale objects. This encompasses 6 DoF frame-to-frame camera tracking and multi-view pose refinement, both based on the SDF-2-SDF energy. Moreover we propose the incorporation of additional geometric and photometric constraints based on properties stored or derived from SDF grids, such as RGB values or normal directions. The related publications are:

- Slavcheva, M., Kehl, W., Navab, N., Ilic, S.: SDF-2-SDF: Highly Accurate 3D Object Reconstruction. In: European Conference on Computer Vision (ECCV) (2016)

- Slavcheva, M., Kehl, W., Navab, N., Ilic, S.: SDF-2-SDF Registration for Real-time 3D Reconstruction from RGB-D Data. International Journal of Computer Vision (IJCV) **126**(6), 615–636 (2017)

Chapter 5 Next, we adapt the SDF-2-SDF strategy to larger volumes and thus present an odometry system working on large industrial objects and indoor spaces via parallel tracking and refinement over partial SDF grids, which we call *limited-extent volumes (LEVs)*. The related work is:

- Slavcheva, M., Ilic, S.: SDF-TAR: Parallel Tracking and Refinement in RGB-D Data using Volumetric Registration. In: British Machine Vision Conference (BMVC) (2016)



(a) Rigid objects.

(b) Non-rigid objects.

Figure 1.2: **Datasets for quantitative evaluation** of rigid and non-rigid reconstruction.

Chapter 6 We now switch focus to the more challenging problem of non-rigid 3D reconstruction from RGB-D input. In this chapter we propose to accomplish this task via gradient flow between SDFs. In addition, we propose two strategies to regularize the flow and compare their advantages, namely AKVF regularization and flow in Sobolev space. The related publications are:

- Slavcheva, M., Baust, M., Cremers, D., Ilic, S.: KillingFusion: Non-rigid 3D Reconstruction without Correspondences. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
- Slavcheva, M., Baust, M., Ilic, S.: SobolevFusion: 3D Reconstruction of Scenes Undergoing Free Non-rigid Motion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

Chapter 7 While all methods described so far operate without explicit data association, this may be needed for tasks such as character animation, texture transfer and video compression. In this chapter we develop two techniques to estimate correspondences between partial RGB-D views based on their lowest-frequency Laplacian eigenfunctions. Part of the work is included in the SobolevFusion approach, while the other related publications are:

- Slavcheva, M., Baust, M., Ilic, S.: Towards Implicit Correspondence in Signed Distance Field Evolution. In: PeopleCap Workshop, IEEE International Conference on Computer Vision (ICCVW) (2017)
- Slavcheva, M., Baust, M., Ilic, S.: Variational Level Set Evolution for Non-rigid 3D Reconstruction from a Single Depth Camera. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) (2018) (under review)

Chapter 8 Finally, we summarize our findings and lay out directions for future research in a concluding chapter.

Appendix A We provide mathematical derivations of the major components of our methods.

Appendix B Here we briefly introduce a correspondence-based non-rigid 3D reconstruction technique in order to appreciate the differences between the two ways to tackle this task. It is based on a probabilistic expectation-maximization variant of non-rigid ICP and relies on patch-based rigidity constraints for deformation regularization.

2 Fundamentals

This chapter gives an overview of the required mathematical background used in the methods developed thereafter. We first introduce the working principle of range sensing devices, and in particular the types used in our work. Next, we explain the relation between depth images and 3D point clouds via projective geometry. Finally, we summarize standard 6 DoF pose parameterizations, common rigid body registration methods, and widespread deformation representations.

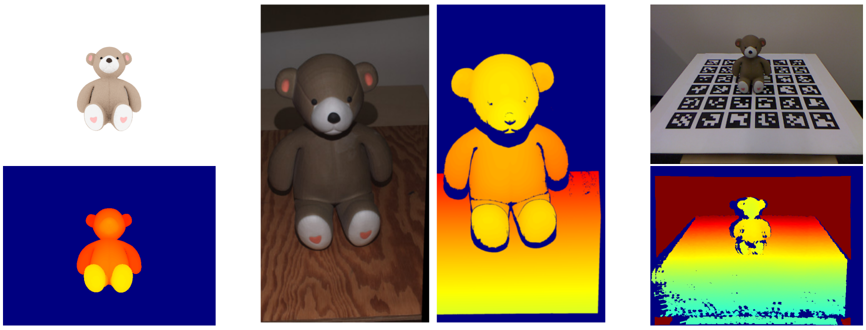
2.1 RGB-D Sensors

A *depth image* is a 2D image acquired by a range-sensing device, which stores the distance to the surfaces observed from the camera center. Given appropriate calibration, each pixel records the distance in physical units. An RGB-D sensor captures a colour image in addition to depth. The images may be of different dimensions and taken with respect to different reference coordinate systems, or, conversely, they may be pre-aligned.

There are various types of 3D scanners, for which the most wide-spread techniques are *triangulation* [83] and *time-of-flight* (ToF) [78].

Triangulation is the process of finding the 3D position of a point given its positions in two images taken from a calibrated set of cameras [83]. It can be either passive, as in stereo vision, or active, as done by *structured light* techniques. Stereo matching is typically accomplished via variants of the PatchMatch algorithm, which carries out random sampling and propagation to surrounding areas in order to find approximate nearest neighbours on the epipolar line according to a plane [11, 17]. Structured light approaches project a known infrared pattern onto the scene and use the pattern distortion, caused by the varying incident depth, to estimate the disparity. *Phase shift* scanners also belong to the family of structured light range-finders [184]. They illuminate the scene with coherent light and use the phase shift of the reflected light relative to the source to deduce depth.

On the other hand, time-of-flight cameras, as the name suggests, measure the time light travels from the emitter to an observed surface and back to the



(a) Synthetic rendering. (b) Industrial phase-shift scanner. (c) Kinect v1.

Figure 2.1: **Sample RGB-D pairs** acquired with some of the sensors used in this dissertation. As the quality degrades from (a) synthetic through (b) industrial to (c) a mass-produced sensor, the depth becomes noisier and more measurements are missing. Blue indicates invalid depth or no depth value.

sensor. A ToF camera captures the entire image using a single light pulse and carries out computations on CMOS integrated circuits or CCD sensors [78]. While the operational principle of a *light imaging, detection, and ranging* (LIDAR) device is similar, it employs a rotating laser beam to gather measurements.

The commercial RGB-D devices which became a commodity in recent years also rely either on structured light, *e.g.* the Microsoft Kinect v1, or time-of-flight, *e.g.* the Kinect v2, to estimate depth [143]. Their quality is reflected through their *depth resolution*, which is the minimal measurable depth difference that can be discerned [116]. It degrades with increasing distance between the camera center and the measured surface. Another metric, the *depth accuracy*, indicates the imprecision in measuring disparity. The differences between the various types of cameras are related to their applicability outdoors, their ability to capture non-Lambertian surfaces, and the quality of the depth data. Some examples are shown in Figure 2.1. Each sensor has its own noise characteristics, both systematic and random [116], which are still difficult to model in general [159]. For example, the Kinect v1 has two operating ranges: near (0.4 - 3.0 m) and far (0.8 - 4.0 m) [2]. It has been determined empirically that the measurement error is already a few millimeters at the start of the range, increasing quadratically up to about 4 cm at its maximum [8, 116]. Recently, learning-based approaches have started emerging in order to both improve the quality of the measured depth, and increase the acquisition speed [64, 65].

2.2 Projective Geometry

Now that we have understood what depth images represent, we consider the image formation process that creates them. We extend this section with some more geometry-related background, including homogeneous coordinates and rigid body motion parameterizations.

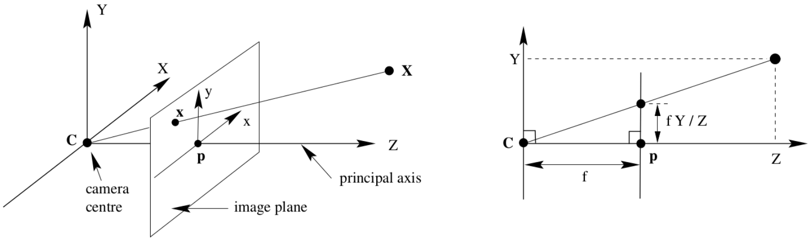


Figure 2.2: **Pinhole camera model.** Image source: *Multiple View Geometry in Computer Vision*, Hartley and Zisserman [84].

Homogeneous Coordinates

In order to conveniently represent operations such as rotation, translation, scaling and perspective projection via matrix-vector products, it is customary to employ homogeneous coordinates [151]. They have one additional coordinate compared to standard Cartesian coordinates used in Euclidean geometry, and are invariant to scaling. To obtain the homogeneous coordinates of a pixel $\mathbf{p} = (p_x, p_y)^\top$, we append a 1, resulting in $\tilde{\mathbf{p}} = (p_x, p_y, 1)^\top$. Similarly, a 3D point $\mathbf{X} = (X, Y, Z)^\top$ is represented by $\tilde{\mathbf{X}} = (X, Y, Z, 1)^\top$ in homogeneous coordinates. To convert back from homogeneous coordinates $(a, b, k)^\top$ we divide by the scale factor, obtaining $(a/k, b/k)^\top$.

Pinhole Camera Model

The projection of a real-world 3D scene onto an image plane is described by the underlying camera model [84]. We use the *pinhole camera model*, which considers the camera as an infinitesimally small hole without any lenses. Light rays which intersect the image plane after passing through the hole obtain a projection on the image, following the process illustrated in Figure 2.2. This model is an ideal approximation, which does not account for distortions for instance. Therefore, it tends to be most accurate around the optical image center and becomes less precise towards its borders. Nevertheless, these limitations can be compensated for with appropriate calibration techniques [222, 255], so the pinhole camera model is extremely widespread in computer vision applications. Among its properties are the fact that straight lines remain straight, while parallel lines intersect at the so-called *vanishing point*.

The *focal length* of a pinhole camera is the distance between the camera center and the image plane. As pixels are generally rectangular, there are two scaling factors for the focal lengths in x- and y-direction respectively, f_x and f_y, measured in pixels. The intersection of the principal axis and the image plane is the *principal point* (c_x, c_y). All of these characteristic quantities are combined into the *intrinsic camera calibration matrix* $\mathbf{K}$:

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} . \tag{2.1}$$

The *projection operator* $\pi\colon \mathbb{R}^3 \to \mathbb{R}^2$ projects a 3D point $\mathbf{X} = (X, Y, Z)^\top$ onto the image plane in the pixel location $\mathbf{p} = (p_x, p_y)^\top$ as follows:

$$\tilde{\mathbf{p}} = \mathbf{K}\begin{pmatrix}\mathbf{I_{3\times 3}} & \mathbf{0_{3\times 1}}\end{pmatrix}\tilde{\mathbf{X}}, \tag{2.2}$$

where $\mathbf{I_{3\times 3}}$ is the 3×3 identity matrix. Thus the pixel coordinates are:

$$p_x = \frac{X}{Z} f_x + c_x\,, \tag{2.3}$$

$$p_y = \frac{Y}{Z} f_y + c_y\,. \tag{2.4}$$

The inverse operation is called *back-projection*. Given pixel coordinates (p_x, p_y) and a depth value Z, for instance known from a depth image, the 3D coordinates are:

$$\left(\frac{x - c_x}{f_x} Z, \frac{y - c_y}{f_y} Z, Z\right)^\top\,. \tag{2.5}$$

The camera might be moved away from the origin by an *extrinsic* transformation consisting of a rotation $\mathbf{R}$ and a translation $\mathbf{t}$, whose properties will be explained in the next section. Then the 3D coordinates $\mathbf{X}$ and the pixel coordinates $\mathbf{x}$ are related by the full *projection matrix*:

$$\tilde{\mathbf{x}} = \mathbf{K}\begin{pmatrix}\mathbf{R} & \mathbf{t}\end{pmatrix}\tilde{\mathbf{X}}\,. \tag{2.6}$$

The parameters of the *intrinsic camera matrix* are determined via calibration techniques and the imaging of known calibration targets [255], while the *extrinsic camera matrix* can be determined via image registration.

Rigid Body Transformations

Rigid body motion is a change in position and orientation, which affects every point of a set in the same way and preserves the relative distance and angle between any pair of points. Thus the set of points remains *rigid*, as for example caused by the motion of a camera imaging a static scene. A rigid body transformation in 3D space consists of a *rotation* and a *translation*, each of which has three degrees of freedom, summing to a total of 6 DoF. All such transformations in 3D Euclidean space form the *special Euclidean group* $SE(3)$ [101].

The *matrix representation* of 3D rigid body motion is the following 4×4 matrix:

$$\mathbf{T} = \begin{pmatrix}\mathbf{R} & \mathbf{t}\\ \mathbf{0} & 1\end{pmatrix}, \tag{2.7}$$

where $\mathbf{R} \in SO(3)$ is a 3×3 orthogonal matrix representing the rotation, and $\mathbf{t} \in \mathbb{R}^3$ is a vector corresponding to the translation. Due to the orthogonality of the rotational component, $\mathbf{R}^{-1} = \mathbf{R}^\top$, the inverse of a rigid body transformation can be computed as:

$$\mathbf{T}^{-1} = \begin{pmatrix}\mathbf{R}^\top & -\mathbf{R}^\top\mathbf{t}\\ \mathbf{0} & 1\end{pmatrix}. \tag{2.8}$$

As there are 16 elements to represent only 6 DoF, it is clear that the matrix notation is an over-parameterization, which is not ideal for optimization. This has lead to the development of various other representations, two of which we describe next and then explain our choice for the remainder of this thesis.

Quaternions are generalizations of complex numbers to 3D. They are represented as 4-element vectors $\mathbf{q} = (q_w, q_x, q_y, q_z)^\top$ with

$$\begin{aligned} \text{norm:} \quad & \|\mathbf{q}\| = \sqrt{q_w^2 + q_x^2 + q_y^2 + q_z^2}\,, \text{ and} \\ \text{inverse:} \quad & \mathbf{q}^{-1} = \frac{\bar{\mathbf{q}}}{\|\mathbf{q}\|}\,. \end{aligned} \tag{2.9}$$

A unit quaternion is a quaternion with unit norm [51]. It is used to represent rotations in 3D space via the following formula [59]:

$$\mathbf{R}(\mathbf{q}) = \begin{pmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_xq_y - 2q_zq_w & 2q_xq_z + 2q_yq_w \\ 2q_xq_y + 2q_zq_w & 1 - 2q_x^2 - 2q_z^2 & 2q_yq_z - 2q_xq_w \\ 2q_xq_z - 2q_yq_w & 2q_yq_z + 2q_xq_w & 1 - 2q_x^2 - 2q_y^2 \end{pmatrix}. \tag{2.10}$$

More specifically, the unit quaternion is a modification of the axis-angle representation, where q_w corresponds to the angle of rotation θ: $q_w = \cos(\theta/2)$, while the remaining elements represent the normalised rotation axis $\hat{\mathbf{r}}$: $(q_x, q_y, q_z)^\top = \hat{\mathbf{r}} \sin(\theta/2)$ [226].

To obtain the quaternion from a rotation matrix, first a solution for one of its four elements has to be determined. To this end, the facts that the quaternion has unit norm is and the rotational matrix is orthogonal are utilized. Afterwards it is straightforward to determine the remaining three elements. The following formulas first calculate q_w, while we refer the interested reader to the technical report of Farrell [66] for the other alternatives:

$$q_w = \frac{1}{2}\sqrt{1 + R_{11} + R_{22} + R_{33}}, \quad \mathbf{q} = \begin{pmatrix} q_w \\ \dfrac{R_{32} - R_{23}}{4q_w} \\ \dfrac{R_{13} - R_{31}}{4q_w} \\ \dfrac{R_{21} - R_{12}}{4q_w} \end{pmatrix}. \tag{2.11}$$

It is essential to normalize to unit quaternions when performing any kind of operations on rotations, so that all other formulas hold true.

Quaternions are popular in robotics, because they allow for convenient interpolation by a factor of μ between two rotations represented as the unit quaternions $\mathbf{q_1}$ and $\mathbf{q_2}$. There are two ways, namely linear interpolation (*lerp*) and spherical linear interpolation (*slerp*) [226]:

$$lerp(\mathbf{q}_1, \mathbf{q}_2, \mu) = (1 - \mu)\,\mathbf{q}_1 + \mu\,\mathbf{q}_2\,, \tag{2.12}$$

$$slerp(\mathbf{q}_1, \mathbf{q}_2, \mu) = \frac{\sin((1-\mu)\alpha)}{\sin\alpha}\mathbf{q}_1 + \frac{\sin(\mu\alpha)}{\sin\alpha}\,\mathbf{q}_2, \;\; \alpha = \arccos(\mathbf{q}_1 \cdot \mathbf{q}_2). \tag{2.13}$$

The disadvantage of *lerp* is that it results in faster movement between 20° and 160°, while *slerp* yields smoother motion, but is more expensive to compute.

In order to represent a full rigid body motion, one has to also account for the translational component, leading to a total of 7 elements in this parameterization - still more than the degrees of freedom. As we target real-time applications and will often solve systems on the GPU, where memory is limited, we look for a representation which is minimal.

Exponential coordinates The Lie algebra $se(3)$ of the $SE(3)$ group provides a way to represent rigid body motion using only 6 elements through *exponential coordinates* [141]:

$$\xi = (\mathbf{u}, \boldsymbol{\omega}) = (u_1, u_2, u_3, \omega_1, \omega_2, \omega_3)^\top , \tag{2.14}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ represents the rotational component of the transformation, while $\mathbf{u} \in \mathbb{R}^3$ corresponds to the translation.

An element of the $se(3)$ algebra has the form

$$u_1\mathbf{G}_1 + u_2\mathbf{G}_2 + u_3\mathbf{G}_3 + \omega_1\mathbf{G}_4 + \omega_2\mathbf{G}_5 + \omega_3\mathbf{G}_6 , \tag{2.15}$$

where $\mathbf{G}_i$ are its generators [58]:

$$\begin{gathered}
\mathbf{G}_1 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \mathbf{G}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \mathbf{G}_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \\
\mathbf{G}_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \mathbf{G}_5 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \mathbf{G}_6 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.
\end{gathered} \tag{2.16}$$

A *twist* $\hat{\xi} \in se(3)$ is a 4×4 matrix parameterized by ξ as follows:

$$\hat{\xi} = \begin{pmatrix} \boldsymbol{\omega}_\times & \mathbf{u} \\ \mathbf{0} & 0 \end{pmatrix}, \tag{2.17}$$

where $\boldsymbol{\omega}_\times$ is the skew-symmetric matrix corresponding to $\boldsymbol{\omega}$:

$$\boldsymbol{\omega}_\times = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} . \tag{2.18}$$

The matrix representation of the respective rigid body motion is obtained via exponentiation:

$$\mathbf{T}(\xi) = \exp\left(\hat{\xi}\right) = \exp\begin{pmatrix} \boldsymbol{\omega}_\times & \mathbf{u} \\ \mathbf{0} & 0 \end{pmatrix}, \tag{2.19}$$

where

$$\exp(\boldsymbol{\omega}_\times) = \mathbf{I} + \boldsymbol{\omega}_\times + \frac{1}{2!}\boldsymbol{\omega}_\times^2 + \frac{1}{3!}\boldsymbol{\omega}_\times^3 + \cdots = \mathbf{I} + \frac{\sin\theta}{\theta}\boldsymbol{\omega}_\times + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\omega}_\times^2 . \tag{2.20}$$

By substituting $\exp(\boldsymbol{\omega}_\times)$ into Eq. (2.19), we obtain a closed-form expression for the conversion from exponential coordinates $\xi = (\mathbf{u}, \boldsymbol{\omega})$ into a transformation matrix [58]:

$$\begin{aligned} \theta &= \sqrt{\boldsymbol{\omega}^\top \boldsymbol{\omega}}, \\ A = \frac{\sin\theta}{\theta}, \quad B &= \frac{1-\cos\theta}{\theta^2}, \quad C = \frac{1-A}{\theta^2}, \\ \mathbf{R} &= \mathbf{I} + A\boldsymbol{\omega}_\times + B\boldsymbol{\omega}_\times^2, \\ \mathbf{V} &= \mathbf{I} + B\boldsymbol{\omega}_\times + C\boldsymbol{\omega}_\times^2, \\ \mathbf{T}(\xi) &= \exp\left(\hat{\xi}\right) = \begin{pmatrix} \mathbf{R} & \mathbf{V}\mathbf{u} \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}. \end{aligned} \tag{2.21}$$

The inverse process is done by taking the logarithm [58]:

$$\begin{aligned} \theta &= \arccos\left(\frac{\operatorname{tr}(\mathbf{R}) - 1}{2}\right), \\ \omega &= \ln(\mathbf{R}) = \frac{\theta}{2\sin\theta}\left(\mathbf{R} - \mathbf{R}^\top\right), \\ \mathbf{u} &= \mathbf{V}^{-1}\mathbf{u} = \left(\mathbf{I} - \frac{1}{2}\boldsymbol{\omega}_\times + \frac{1}{\theta^2}\left(1 - \frac{A}{2B}\right)\boldsymbol{\omega}_\times^2\right)\mathbf{t}. \end{aligned} \tag{2.22}$$

Next, we derive the Jacobian of a point in 3D Euclidean space with respect to the twist which generated it. Let $\mathbf{X} = (x_1, x_2, x_3)^\top$ be a 3D point to which a rigid body transformation $\mathbf{T}$ is applied, moving it to point $\mathbf{Y} = (y_1, y_2, y_3)^\top$. In homogeneous coordinates:

$$\widetilde{\mathbf{Y}} = \mathbf{T}\widetilde{\mathbf{X}} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \widetilde{\mathbf{X}}. \tag{2.23}$$

Using the generators of the Lie algebra, we rewrite the above equation as:

$$\begin{aligned} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ 1 \end{pmatrix} &= \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} = \\ &= (u_1\mathbf{G}_1 + u_2\mathbf{G}_2 + u_3\mathbf{G}_3 + \omega_1\mathbf{G}_4 + \omega_2\mathbf{G}_5 + \omega_3\mathbf{G}_6) \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} = \\ &= \begin{pmatrix} u_1 & & & + & \omega_2\, x_3 & - & \omega_3\, x_2 \\ u_2 & - & \omega_1\, x_3 & & & + & \omega_3\, x_1 \\ u_3 & + & \omega_1\, x_2 & - & \omega_2\, x_1 & & \\ & & & & 0 & & \end{pmatrix}. \end{aligned} \tag{2.24}$$

It is now straightforward to calculate the Jacobian by deriving with respect to each of the six exponential coordinates:

$$\begin{aligned}\frac{\partial \mathbf{Y}}{\partial \xi} &= \begin{pmatrix} \frac{\partial y_1}{\partial u_1} & \frac{\partial y_1}{\partial u_2} & \frac{\partial y_1}{\partial u_3} & \frac{\partial y_1}{\partial \omega_1} & \frac{\partial y_1}{\partial \omega_2} & \frac{\partial y_1}{\partial \omega_3} \\ \frac{\partial y_2}{\partial u_1} & \frac{\partial y_2}{\partial u_2} & \frac{\partial y_2}{\partial u_3} & \frac{\partial y_2}{\partial \omega_1} & \frac{\partial y_2}{\partial \omega_2} & \frac{\partial y_2}{\partial \omega_3} \\ \frac{\partial y_3}{\partial u_1} & \frac{\partial y_3}{\partial u_2} & \frac{\partial y_3}{\partial u_3} & \frac{\partial y_3}{\partial \omega_1} & \frac{\partial y_3}{\partial \omega_2} & \frac{\partial y_3}{\partial \omega_3} \end{pmatrix} = \\ &= \begin{pmatrix} 1 & 0 & 0 & 0 & x_3 & -x_2 \\ 0 & 1 & 0 & -x_3 & 0 & x_1 \\ 0 & 0 & 1 & x_2 & -x_1 & 0 \end{pmatrix} = \begin{pmatrix} \mathbf{I}_{3\times3} & -\mathbf{X}_\times \end{pmatrix} . \end{aligned} \tag{2.25}$$

A major benefit of the Lie algebra representation is the fact that by definition it is a vector space with a special operation called the *Lie bracket* [79], and is thus closed under scalar multiplication. This property is essential for numerical approaches, such as gradient descent, as it permits the multiplication with a scalar step size, guaranteeing that the result will be a rigid body motion. This is contrary to transformation matrices, where it is not certain whether the orthogonality of the rotation matrix will be conserved.

2.3 Registration

In this section we discuss the most widely used approaches for registration in static environments, as well as commonly used models for non-rigid tracking.

2.3.1 Rigid Motion

Given point clouds obtained from different points of view upon the surface of a static object, the *registration task* is to place them into a common reference frame by estimating the relative rigid body transformation between them [37, 71]. The problem can be further classified into *pairwise registration*, when there are only two views, and *multi-view registration*, when there are more. If an initial guess about the transformation is available, we only need to carry out *registration refinement*. In the absence of any prior knowledge, we talk about *unconstrained registration* [96]. As the registration problem is fundamental for many computer vision tasks, there is long-standing research on all of these scenarios.

Iterative Closest Points (ICP) is arguably the most common approach for rigid point cloud registration. It was introduced in the early 1990s almost simultaneously by Besl and McKay [15] and Chen and Medioni [37], but is subject to improvements even nowadays [259]. The earliest algorithms could cope with moderate amounts of normally distributed noise [15], but could not perform well in the presence of gross statistical outliers and were very computationally demanding. Therefore, many extensions followed, which are summarized in the comprehensive overview by Rusinkiewicz and Levoy [180].

The steps of any ICP variant are the following: *point selection* in one or both datasets; *matching* between the selected points; *weighting* the point correspondences; *rejection* of unreliable pairs; and *minimisation* of a chosen error metric [180].

An important remark is that the classical ICP approach as suggested by Besl and McKay is *guaranteed to monotonously converge to a local minimum from any initial setting*, although this might not be the global minimum [15]. However, extensions often do not even have a proof of convergence to a local minimum [254]. Therefore, there is a trade-off between the robustness and convergence properties of any ICP variant. We consider some of the most prominent of them next.

The issues of the earliest two approaches, namely sensitivity to noise and slowness, were addressed shortly after. While Besl and McKay used a point-to-point error metric between the datasets, Chen and Medioni proposed a point-to-plane measure, which is usually more accurate [188]. To increase robustness with respect to occlusion and outliers, Zhang [254] analyzed the distance distribution, deriving a statistical method for outlier rejection. Fitzgibbon [71] tackled the runtime limitations and suggested a speedup by employing the distance transform representation. Extensions to multiview settings were already outlined by Chen and Medioni as a global view-to-model process, and followed shortly after [145, 155, 170].

Johnson and Kang [104] also addressed the multiview registration problem via an ICP modification, which considers not only 3D information, but also colour, called *colour ICP*. Henry *et al*. [86] proposed a similar approach for RGB-D settings, called *RGBD-ICP*, which additionally employs the point-to-plane metric. Colour integration has also been demonstrated as advantageous in multiple examples in Bernardini and Rushmeier's overview of the 3D model acquisition pipeline with range data registration [13]. To sum up, photometric constraints have the potential to significantly decrease the registration error when sufficient texture is available.

Similarly, Schütz *et al*. [186] developed another ICP extension, which utilizes not only colour, but also surface orientation constraints by evaluating the consistency between normal vectors. Their *multi-feature ICP* therefore copes with ambiguous cases that lack prominent geometric features. The authors demonstrate that if either colour or normal constraints are added to the geometric error term on their own, they aid registration, while all three components together yield even more precise results.

In conclusion, the ICP algorithm is general, simple and extensible, but has numerous failure cases. Even though there exists a closed-form solution if the point-to-point metric is employed and quaternions are used to represent rotation [91], iterative solvers are required in case of noisy or missing data. For instance, Besl and McKay point out that brute-force comparisons exhausting all possible point correspondences might take several universe lifetimes to find the best match, while their ICP implementation reduces the computation time by many orders of magnitude, but runtime is still an issue [15]. Therefore, approaches which circumvent the computation of all explicit point matches are of great research interest.

Frame-to-model ICP variants *KinectFusion* [157] is a work that inspired significant progress in 3D reconstruction from RGB-D sensors, because it was the first to demonstrate dense tracking and mapping of an indoor scene in real time. It is based on an ICP variant that aligns the points from an incoming depth image with a predicted view generated from the whole current reconstruction, *i.e.* it is a frame-to-model alignment strategy. More precisely, there is a global TSDF model, which is incrementally updated after the pose of each frame is determined using the volumetric surface integration framework of Curless and Levoy [45]. The TSDF is ray-casted into a noise-free vertex map, which is used for registration and for display to the user. While the frame-to-model approach benefits from the fact that the global reconstruction is continuously refined as more measurements are integrated, if very incorrect poses are used for fusion, the tracking of all subsequent frames will suffer. Moreover, as no additional refinement is employed, accumulated drift cannot be compensated and the final model cannot be further improved. This limitation has been subsequently addressed in BundleFusion [46] which re-integrates erroneously fused surface components.

Point-to-implicit ICP variants circumvent the explicit matching step of ICP via direct alignment between the incoming point cloud and the zero level set of the global TSDF, *i.e.* they register a point cloud to an implicit surface representation [177]. The approach of Kubacki *et al.* [120, 121] proposes a novel ICP matching criterion and an error metric based on the properties of implicit representations. Canelhas *et al.* [34, 35] and Bylow *et al.* [30] minimize the sum of squared point-to-model distances. All of these methods use iterative solvers based on a Taylor linearisation of the objective function to build the 6×6 system for 6 DoF pose updates. Paragios *et al.* [165] suggest an alternative to this strategy by determining the Jacobian of the registration energy with respect to the camera pose, so that a simpler gradient descent scheme can be carried out. While these approaches tend to be faster and more robust than classical ICP, they are imprecise if an incoming point cloud consists of very few measurements or if the initial pose guess is very far from the optimum, so even denser approaches are of interest to increase accuracy.

Direct registration methods offer a different way to avoid ICP-like correspondence search. Instead, they minimize an error over the entire image, for example taking intensity difference as the metric [100]. They have been investigated in the contexts of visual odometry [115, 224] and SLAM [158]. These approaches have an advantage over feature-based or sparse ICP variants in the case of poorer geometry or under rapid motion, as the information from the whole image can stabilize pose estimation.

Pose optimizaiton Regardless of the initial registration technique that has been employed, many methods carry out a final pose optimization step that uses the information from multiple views jointly in order to improve the estimated trajectory. Nevertheless, if the initial error is too large, all techniques would get stuck into local minima [191].

A common approach for global refinement is to select a set of *keyframes* that build a graph structure and perform *graph optimization* [122]. The nodes are the keyframe camera poses, while the edges are the transformations between them. The objective is to determine the transformations which yield the smallest global misalignment, expressed by the strongest multiview geometry consensus. While many authors have successfully employed this strategy [52, 111, 114, 178, 191, 203], the number of connections in the graph increases exponentially and thus sets a limit to its applicability in online systems. Due to this issue, graph-free approaches are subject to investigation. For example, the task can be accomplished via surface deformations instead of changing the previously determined camera poses [239, 258].

2.3.2 Non-rigid Motion

Having considered some of the major rigid registration techniques, we now focus on their modifications that allow handling non-rigid movements.

In addition to facing the same challenges as in static environments, such as limited overlap, missing and noisy data, the *non-rigid registration* problem is significantly underconstrained since infinitely many mappings may deform one shape into another [74]. This is easy to imagine even in a lower dimension, since if only two points are given, any trajectory could displace one of them to the the other. Taking inspiration in physical phenomena, researchers impose various constraints in order to reduce the solution space and find the most plausible motion. Thus typically they optimize an energy function composed of *data terms* that enforce alignment between the two shapes, and *regularizers* that are based on assumptions about the possible solutions.

Non-rigid ICP aims to adapt the classical ICP algorithm to the dynamic case [7, 26, 67]. The task is defined as finding a *warp field* that brings the shapes in optimal alignment, while estimating the point correspondences between them. This joint problem is typically tackled via an expectation-maximization (EM) [50] procedure. In the E-step the correspondences are updated based on the current warp estimate, and in the M-step the field is updated given these correspondences, after which the optimization is repeated until convergence [263]. Thanks to splitting the complexity of the problem into these two simpler steps, many state-of-the-art non-rigid 3D reconstruction systems are based on non-rigid ICP, and some even achieve real-time performance [31, 55, 80, 94, 98, 132, 156].

Volumetric deformation techniques are inspired by the *free-form deformation* (FFD) framework [187] which is driven by the following analogy: the deformable object is imagined to be embedded in a clear, flexible volume of plastic, populated with a set of control points. These control points are then displaced and the position of any other point is determined with the aid of a tensor product trivariate Bernstein polynomial, giving rise to a deformed shape.

Similarly, deformation based on regular voxel grids is often used in medical imaging, both for volume registration and segmentation [125, 144, 260]. Often the employed representation is a level set, which is evolved [9, 87]. In computer vision, Paragios *et al.* [165] use distance functions for 2D non-rigid registration driven by a vector field, while Fujiwara *et al.* [74] demonstrate their locally rigid, globally non-rigid dual-grid FFD framework between two SDFs on both 2D and synthetic 3D examples. The advantage of these approaches is not only that they do not entail correspondence estimation, but that they also inherently cope with changes in topology.

Some techniques couple a regular volumetric grid representation with correspondence-driven registration [98, 262]. Even though they are not able to cope with topological changes, they benefit from the regularity which makes them very suitable for GPU parallelization and therefore for real-time processing.

Deformation regularization is crucial to reduce dimensionality and make the non-rigid registration problem tractable [74]. Some techniques employ *multi-view* constraints [4, 31, 230], while others resort to prior knowledge such as rigidly acquired *templates* [95, 261] or *parametric models* of hands [214], faces [219] or entire bodies [19, 21] with embedded skeletons [229, 75]. These strategies are, however, restricted to a specific class of surfaces or require recording in specialized studios. Therefore general regularization through terms in the minimized non-rigid energy has been subject to extensive research.

Linear regularization methods formulate surface deformation as a variational optimization problem and linearize the energy functional, obtaining a linear equation system which can be efficiently solved [23]. A commonly used model is that of thin-plate splines [22, 25, 26]. However, these techniques do not cope well with rotational motion and are thus mainly used only for fine-scale refinement after an approximate solution is already available [263].

Non-linear regularization methods are more descriptive of the underlying deformation process and are therefore used in many state-of-the-art non-rigid reconstruction systems. Most of them are driven by the intuition that surfaces do not deform randomly, but have some physical constraints that make them stay close to rigid. This is especially valid for the case of moving humans, since their motion is only articulated, as opposed to truly deformable such as an expanding balloon.

One of the most widely employed regularization frameworks is the *as-rigid-as-possible* (ARAP) model [201]. It aims to preserve the first and second fundamental forms of the surface, which are related to its extrinsic invariants such as principal curvatures and metric distances. Therefore the resulting deformation is locally as-rigid-as-possible, preventing physically unlikely configurations caused by excessive stretching or sheering. It has been used both in template-based [261] and template-free [98] dynamic reconstruction methods.

Approximately Killing vector fields (AKVFs) [12, 200, 215] enforce a similar effect directly through constraints over the warp. They impose antisymmetry over the Jacobians of the field and thus divergence-free behaviour, which generates locally nearly isometric deformations. Thus they minimize an ARAP energy to first order [200].

Another frequently used paradigm is *embedded deformation* (ED) [209], which, in addition to local rigidity, also imposes spatial smoothness of the deformation field. It is represented as a graph structure in which the surface geometry is embedded. Each graph node is associated with a transformation, which affects the movements of those parts of the shape that are located in nearby space, leading to an extremely powerful model. The edges between them indicate local dependencies and enforce global consistency of the overall deformation. The optimal state is found via a non-linear minimization that determines the values of the node transformations. The majority of correspondence-based non-rigid reconstruction systems rely on an ED graph [56, 55, 81, 132, 156, 247].

In the following we will take inspiration from all discussed techniques and aim to mitigate their drawbacks via the new methods that we propose.

Part II

Signed Distance Field Alignment

3 Direct SDF Alignment Energy

The main representation that we will explore for the development of our new 3D reconstruction methods in this thesis is the *signed distance field*. This chapter is devoted to explaining its properties, the way we generate it in the discrete setting of a digital implementation, and the intuitions behind the energy that will be used to align pairs of SDFs.

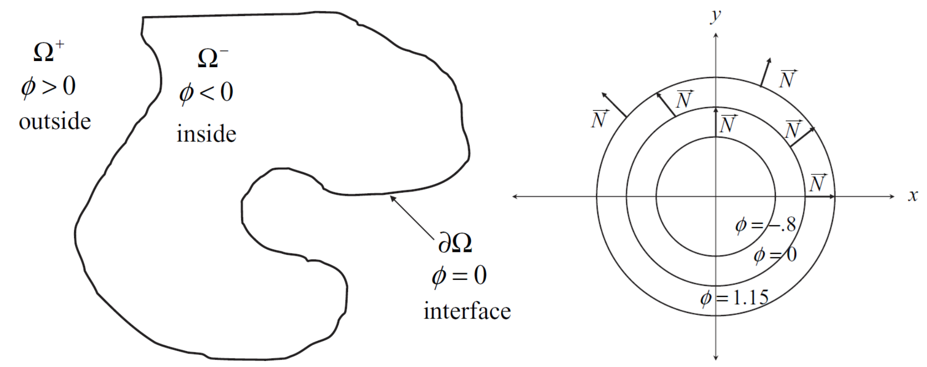
Signed distance fields have a broad range of applications in computer vision, graphics and medical imaging. They have been used for curve smoothing, detection of dominant points on curves, finding convex hulls, determining object skeletons, centerlines and medial axes, computing Dirichlet tessellations, morphing, hypertexture, scene motion, collision detection, obstacle avoidance, and many others [106, 152, 246]. They are also used for efficient multi-sensor fusion, such as for combining the information from sonar and stereo [60].

3.1 SDF Definition

A signed distance function is an n-dimensional *implicit function*, which associates a scalar value with each point of its n-dimensional domain [163]. As our objective is 3D reconstruction, we will primarily deal with 3-dimensional space and assume $n = 3$ from here onwards, unless stated otherwise. Formally, the function

$$\phi \colon \Omega \subseteq \mathbb{R}^3 \to \mathbb{R} \tag{3.1}$$

assigns to each point in 3D space $\mathbf{X} \in \mathbb{R}^3$ its signed distance to the closest object boundary, *i.e.* to the nearest surface location. Points located within the object bounds have negative signed distance values, while points outside are positive-valued, as indicated in Figure 3.1a. Therefore the zero-valued interface between them *implicitly* defines the object surface. Its explicit counterpart, such as a mesh representation, can be extracted via methods such as marching cubes [136] and ray tracing [6, 166].



(a) Positive-, negative-, and zero-valued SDF regions.

(b) SDF gradient.

Figure 3.1: **Signed distance fields in 2D** indicating (a) the positive-valued outside, negative-valued inside and zero-valued interface; as well as (b) the SDF gradient, which is always orthogonal to the level sets. Image source: *Level Set Methods and Dynamic Implicit Surfaces*, Osher and Fedkiw [163].

SDFs have many advantages over other boundary representations. They not only represent the surface, but also its interior and its surrounding volume, which can be utilized in registration tasks. Moreover, they provide an inexpensive means to compute any offset surface by simply changing the extracted level set value [73]. Note that for $n = 3$ the level sets are called *isosurfaces*, while for $n = 2$ they are called *isocontours*.

However, implicit functions are very costly to compute [73]. This is why they are usually represented as discrete volumes, subdivided into voxels. With this limitation in mind, from here on we will purposefully avoid using the term *signed distance function* when referring to the digital implementation, since it is discrete and the actual underlying function is typically not known analytically. What is being dealt with is a 3D voxel grid of scalar values, which, in our opinion, is better described by the term *field*, even though both expressions are abbreviated as *SDF*.

We choose to use cubic voxels, while other shapes can also be utilized. As the *voxel size* approaches zero, it approximates a point better and better, and therefore the discrete SDF becomes closer to the actual distance function. However, the available memory is a limiting factor for the choice of voxel dimensions, especially when stored on a GPU with restricted global memory. Consequently, it also influences the quality of the approximation that the discrete grid provides for the underlying SDF.

3.2 SDF Properties

Before delving into the algorithmic properties of SDF generation and alignment, we take a moment to describe some of the most important geometric properties of SDFs. They will help us in choosing appropriate energy terms in our methods.

SDF gradient One of the characteristic properties is the fact that the *SDF gradient is orthogonal to the isosurfaces* everywhere in the volume. Moreover, its *magnitude is one*, since distance is a Euclidean measure, and thus moving twice as close to the surface from a given point in space results in a signed distance value which is two times smaller [163, 106]. Therefore, at the surface the SDF gradient equals the *unit surface normals*, as visualized in Figure 3.1b. Mathematically:

$$\|\nabla_{\mathbf{x}}\, \phi(\mathbf{X})\| = \|\overline{\mathbf{n}}(\mathbf{X})\| = 1, \qquad \forall \mathbf{X} \in \partial\Omega. \tag{3.2}$$

Above we use the symbol $\nabla_{\mathbf{x}}$ to refer to the spatial gradient of the SDF ϕ. In addition, $\overline{\mathbf{n}}$ denotes the unit surface normal.

While Eq. (3.2) is valid in continuous space, it is not defined at points which are equally distant from more than one surface location, such as the center of a sphere. In such cases the gradient is undefined [106].

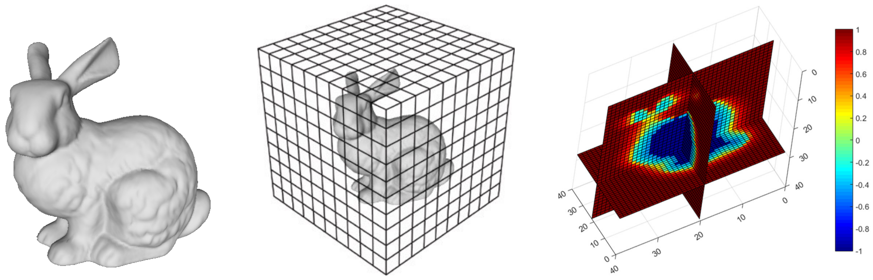
On the contrary, in a discrete setting the gradient is calculated via a numerical scheme, *e.g.* finite differences, so it is defined everywhere. However, because of loss of accuracy due to voxel disretization, the norm might not have unit magnitude any more [163]. Therefore many numerical implementations re-normalize it in a process called *SDF re-initialization*. It is achieved either through imposing a partial differential equation as a hard constraint at selected steps in a given algorithm [130], or as a soft constraint through an energy term that enforces the gradient to have unit magnitude [129]. This ensures that the SDF is valid and all of its properties are preserved.

Viewpoint independence A signed distance field is viewpoint-independent because the shortest distances to the surface do not depend on where they are viewed from. However, in a discrete setting this holds true only for SDFs representing the entire object after multiple views are fused together. When the SDF is incomplete, as obtained from a single or very few frames, certain voxels have not been observed yet and have to be excluded from computations. This is why single-frame SDFs, as for example generated from a single depth image, are referred to as *projective SDFs*.

3.3 SDF Generation

Next, we move to the discrete setting and describe the SDF generation process used in our methods. At this stage we consider the volume as a regular voxel grid, while it is possible to use memory-efficient representations such as octrees [92, 252], hierarchical structures [36] or hashed volumes [108, 161].

An RGB-D pair consists of a colour image $I_{RGB} : \mathbb{N}^2 \to \mathbb{R}^3$ and an aligned depth map $I_D : \mathbb{N}^2 \to \mathbb{R}_0^+$. As discussed, a single depth image allows us to generate a *discrete projective truncated SDF*. For this purpose, first its bounding volume is determined by back-projecting all pixels to 3D, determining the maximum extents in each spatial direction, and adding some slight padding to allow for movement if the camera pose is changed. Then the volume is



(a) Surface view. (b) Bounding volume. (c) Generated SDF.

Figure 3.2: **Discrete signed distance field generation**: (a) view of the surface of interest; (b) bounding volume discretization; (c) voxel grid with SDF values outlining distinct level sets.

discretized into cubic voxels of a predefined side length l, as outlined in Figure 3.2.

A point $\mathbf{X}$ lies in the voxel with index $vox : \mathbb{R}^3 \to \mathbb{N}^3$:

$$vox(\mathbf{X}) = int\left((\mathbf{X} - \mathbf{C})/l - (1/2, 1/2, 1/2)^\top\right) , \tag{3.3}$$

where $int(\cdot)$ is an operator that rounds to integers, and $\mathbf{C}$ is the lower-left corner of the volume. All points within the same voxel are characterized by the same properties as its center

$$\mathbf{V}(\mathbf{X}) = l(vox(\mathbf{X}) + (1/2, 1/2, 1/2)^\top) + \mathbf{C} , \tag{3.4}$$

thus we denote the entire voxel by $\mathbf{V} \in \mathbb{R}^3$.

Since a depth image only stores measurements of surface points, the projective signed distance is the difference of the sensor reading for the voxel center projection $\pi(\mathbf{V})$ and its depth $\mathbf{V}_Z$. This leads to the following *SDF generation procedure*:

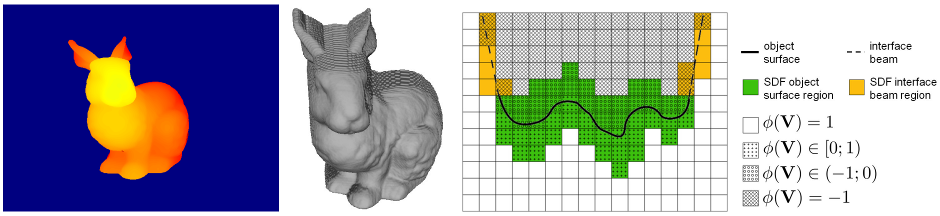
$$d(\mathbf{V}) = I_D(\pi(\mathbf{V})) - \mathbf{V}_Z , \tag{3.5}$$

$$\phi(\mathbf{V}) = \begin{cases} sgn(d(\mathbf{V})) & \text{, if } |d(\mathbf{V})| \geq \delta \\ d(\mathbf{V})/\delta & \text{, otherwise} \end{cases} \tag{3.6}$$

$$\omega(\mathbf{V}) = \begin{cases} 1 & \text{, if } d(\mathbf{V}) > -\eta \\ 0 & \text{, otherwise} \end{cases} \tag{3.7}$$

$$\zeta(\mathbf{V}) = I_{RGB}(\pi(\mathbf{V})) . \tag{3.8}$$

Here $d(\mathbf{V})$ is the view-dependent projective distance, while $\phi(\mathbf{V})$ is the value that we store. The viewpoint-dependence effect is diminished by scaling the values by a factor δ and truncating them to the interval $[-1, 1]$, so that erroneous far-away measurements are disregarded. Similarly, as only values near the object boundary are of interest for surface reconstruction methods, a common speed-up practice is to execute calculations only in a *narrow band* near it [3, 137, 242]. The chosen value of $\delta > 0$ determines its extent, while the binary check $|\phi(\mathbf{V})| < 1$ verifies which voxels belong to it.


object surface
interface beam
SDF object surface region
SDF interface beam region
$\phi(\mathbf{V}) = 1$
$\phi(\mathbf{V}) \in [0;1)$
$\phi(\mathbf{V}) \in (-1;0)$
$\phi(\mathbf{V}) = -1$

(a) Depth map. (b) *Beams*. (c) Cross section along the $x - y$ plane.

Figure 3.3: **Single-frame projective truncated SDF**: (a) depth map; (b) rendering of the respective marching cubes result, showing *interface beams* between regions of 1s and -1s; (c) cross section identifying different parts of the volume.

The binary weight $\omega(\mathbf{V})$ indicates whether the signed distance value for a voxel is reliable. All visible locations and a region of size $\eta > 0$ behind the surface, reflecting the expected object thickness, are assigned weight one. Voxels with zero weight are discarded from computations.

The values of δ and η are somewhat object- and method-dependent. We typically use δ of about 3-10 voxel sizes and η of about 2-5 voxel sizes.

Finally, we store the RGB triple corresponding to each voxel in another grid, ζ, of the same resolution as ϕ. Note that colour is meaningful only near the surface, but it can be propagated in normal direction in order to populate the entire grid with values.

The outlined single-frame SDF generation approach creates *interface beams* where the camera rays pass the surface silhouette, because values of 1 and -1 are adjacent there, as shown in Figure 3.3b. As beams are viewpoint-dependent, we favour SDF re-generation over interpolation when the camera pose is re-estimated. They cancel out when multiple SDFs are fused, but have faulty gradients that need to be omitted from calculations. This is easily done, since the central difference gradient on a beam has at least one component with absolute value 1, and since voxels behind the surface have not been observed and thus have zero weight.

SDF Fusion of several SDFs from different viewpoints is done via the rolling weighted average approach of Curless and Levoy [45]:

$$\begin{aligned} \Phi_{t+1}(\mathbf{V}) &= \frac{W_t(\mathbf{V})\Phi_t(\mathbf{V}) + \omega_{t+1}(\mathbf{V})\phi_{t+1}(\mathbf{V})}{W_t(\mathbf{V}) + \omega_{t+1}(\mathbf{V})}\,, \\ W_{t+1}(\mathbf{V}) &= W_t(\mathbf{V}) + \omega_{t+1}(\mathbf{V})\,. \end{aligned} \tag{3.9}$$

This is a formula for volumetric updates, where Φ_t is the cumulative SDF up to frame number t, while W_t is the respective cumulative weight. It may happen that certain voxels have not been observed from any viewpoint, so their cumulative weight is zero. This is likely to occur only for voxels which are inside the object. Therefore, for implementation purposes their signed distance is directly set to -1 and division by zero is avoided.

Colour fusion can be done similarly by considering each channel of the RGB grid separately and applying Eq. 3.9 over it. However, as noted by Bylow *et al.* [30, 29], it is more accurate to weight colour according to the deviation θ of the line of sight from the surface normal, in addition to the weight of the voxel itself. This strategy assigns larger certainty to the colours of points whose normal is pointing towards the camera. The equations below summarise this procedure, where w^c is the colour weight of a voxel, which is the same for all three channels. The superscript j refers to each of the R, G and B channels, C_t is the cumulative colour grid at frame t, and W_t^C is the corresponding cumulative colour weight:

$$\begin{aligned} w^c_{t+1}(\mathbf{V}) &= \omega_{t+1}(\mathbf{V})\cos(\theta_{t+1}(\mathbf{V}))\,, \\ C^j_{t+1}(\mathbf{V}) &= \frac{W^C_t(\mathbf{V})C^j_t(\mathbf{V}) + w^c_{t+1}(\mathbf{V})\zeta^j_{t+1}(\mathbf{V})}{W^C_t(\mathbf{V}) + w^c_{t+1}(\mathbf{V})}\,. \end{aligned} \tag{3.10}$$

3.4 Energy Formulation

Finally, we introduce the alignment energy between two signed distance fields. It starts from the observation that ICP seeks to match the surface points of two shapes, while point-to-implicit approaches employ surface points on one side and an entire volume on the other side, *i.e.* they make use of the entire shape to steer towards better overlap. The benefit of using the whole shape rather than only the surface has been pointed out in other reconstruction and tracking frameworks too [94]. Our reasoning is to go one step further and instead use both shapes in their entirety, including their inside and outside regions, to improve alignment.

This intuition is illustrated via a 2D analogy of rigid reconstruction methods that use an SDF representation for a given purpose in Figure 3.4. Solid lines correspond to the zero level set of an SDF, which has positive values on one side, and negative values on the other. Note that for our approach each voxel contributes to one summand in the energy, but we visualize it more densely to highlight the fact that, as opposed to point clouds, SDFs have values everywhere in the volume.

In any pair-wise alignment there is a *reference shape* and a *data shape* that has to be fit to the target. In our methods the reference $\phi_{reference}$ will be either the cumulative SDF or the projective truncated SDF of the last tracked frame, while the data ϕ_{other} will typically be the current projective truncated SDF. The SDF-2-SDF *impicit-to-implicit alignment energy* minimizes the sum of squared voxel-wise differences of a pair of SDFs that occupy the same volume:

$$E_{implicit}(\Upsilon) = \frac{1}{2}\sum_{voxels}\left(\phi_{reference} - \phi_{other}(\Upsilon)\right)^2\,, \tag{3.11}$$

where Υ is the objective we are looking for. In the case of rigid reconstruction it will be the 6 DoF pose ξ, while in the case of non-rigid reconstruction it will be a dense deformation field Ψ, as discussed in the respective chapters later.

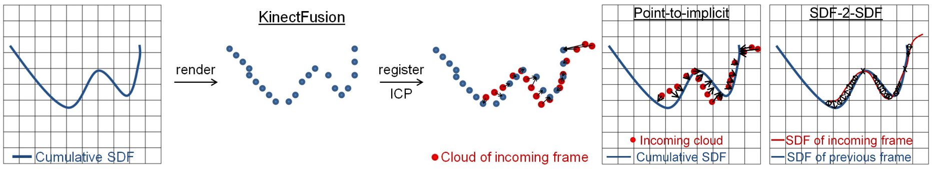


Figure 3.4: **2D analogy for comparison between the operational principles** of KinectFusion [157], point-to-implicit [30, 35] and our proposed SDF-2-SDF.

The energy might also take the SDF weight fields into account in order to exclude unobserved voxels:

$$E_{weighted\ implicit}(\Upsilon) = \frac{1}{2}\sum_{voxels}\left(\phi_{reference}\omega_{reference} - \phi_{other}(\Upsilon)\omega_{other}(\Upsilon)\right)^2 . \quad (3.12)$$

This energy formulation has an advantage over ICP in that it is a direct difference without need for correspondence estimation - upon optimal alignment voxels with the same indices in either volume will correspond to each other. The benefit over point-to-implicit strategies is that our formulation is symmetric and will yield nearly identical results if the target and data volumes are swapped. On the contrary, the point-to-implicit result would depend on which one is represented as a cloud. In particular, if an incoming frame is very noisy and consequently its 3D cloud is very corrupted, registration can be significantly impaired. Thanks to the smoothing properties of implicit functions, our SDFs are likely to be less influenced by noise.

Last but not least, we make use of the truncated ±1s, as opposed to other methods that simply designate them as empty space, often in order to reduce storage requirements [109, 161]. Even though this strategy constrains us to a regular voxel grid structure for the time being, it increases the number of sample points and ensures that convergence from a larger initial deviation is possible.

Thus we attribute our energy function design choice to the geometric benefits of SDFs, including dense sampling of space, a meaningful gradient, and lower sensitivity to noise.

Part III

Rigid 3D Reconstruction

4 SDF-based Object Reconstruction

The first application that we tackle with our implicit-to-implicit energy is precise 3D reconstruction of small- and medium-scale objects, such as household items. Scanning is usually executed with an inward-facing turntable or hand-held trajectory around the object of interest, which is clearly visible without any obstructions, apart from possible self-occlusions. This is typical for domains such as non-destructive testing of industrial machines and components, or in the acquisition of datasets for robotic grasping and manipulation. Hence our method for 3D object reconstruction from a single RGB-D sensor has to be:

- fast;
- fully automatic;
- highly accurate;
- able to handle generic geometry and texture;
- robust to generic motion.

In the following we present our strategy to fulfil these requirements. Our main contribution is a novel *implicit-to-implicit registration scheme between signed distance fields*, called SDF-2-SDF, which we apply both for real-time frame-to-frame camera tracking and for posterior frame-to-model global optimization. SDF-2-SDF alignment is a *direct voxel-wise difference* minimization that circumvents the computationally expensive correspondence search employed by other pose estimation methods. Moreover, it allows for straightforward incorporation of additional geometric and photometric constraints over voxel grids, yielding highly accurate 3D models. An extensive quantitative evaluation demonstrates improved tracking and higher fidelity reconstructions than a variety of state-of-the-art systems. Last but not least, we create a publicly available 3D-printed object reconstruction dataset, which is the first to include ground-truth CAD models and RGB-D sequences from sensors of various quality.

4.1 Introduction

Recovering the geometry of a static object from a moving camera entails estimating the device motion and fusing the acquired depth images into consistent 3D models. Depending on the objective, methods differ in their speed, accuracy and generality. Most existing solutions are derived from simultaneous localization and mapping (SLAM) techniques, thus their applications lie in the field of robotic navigation where precise reconstructions are of secondary importance. In contrast, the growing markets of 3D printing, reverse engineering, industrial design, and object inspection require rapid prototyping of *high quality models*, which is the aim of our system, as shown in Figure 4.1.

KinectFusion [157] is one of the most influential works capable of real-time tracking and reconstruction. It conveniently stores the recovered geometry in an incrementally built signed distance field, which is continuously refined as more measurements are fused in. However, the employed frame-to-model ICP [15, 37] approach to camera tracking limits it to objects with distinct geometry and to uniform scanning trajectories. Alternative volumetric techniques employ a point-to-implicit scheme [30, 35]. It avoids explicit correspondence estimation by directly aligning the point clouds of incoming depth frames with the zero level set of the growing SDF. While this strategy has shown higher robustness than ICP, it becomes unreliable when range data is sparse or once the global model starts accumulating errors.

Dense visual odometry (DVO) [115] combines image intensities with depth information for registration via whole image warping between RGB-D frames. Although it is susceptible to drift on poorly textured scenes, DVO achieves impressive accuracy in real time and has been incorporated as the tracking component of many subsequent systems, including the object reconstruction pipeline of Kehl *et al.* [111]. The final step of the latter is a g^{2}o pose graph optimization [122] that ensures optimal alignment between all views. While it improves the geometry of the final model, it might become prohibitively expensive for a large number of keyframes.

Addressing these limitations, we develop a system for highly accurate 3D object reconstruction, named SDF-2-SDF. It comprises online frame-to-frame camera tracking, followed by swift multi-view pose optimization during the generation of the output reconstruction. Both of these stages employ our SDF-2-SDF registration method, which directly minimizes the difference between pairs of SDFs. Moreover, its formulation allows for integration of surface colour and normal information for even better alignment. In addition to handling larger motion, our frame-to-frame tracking strategy avoids drift caused by errors in the global model. Finally, our global refinement is faster than the pose graph optimization used in other pipelines [52, 111, 122]. Tackling the lack of a dataset combining ground-truth CAD models and RGB-D sequences with known camera trajectories, we acquire such test data and make it publicly available[1]. We summarize these contributions as follows:

- precise implicit-to-implicit registration between SDFs for online frame-to-frame camera tracking;

[1] `http://campar.in.tum.de/personal/slavcheva/3d-printed-dataset/index.html`

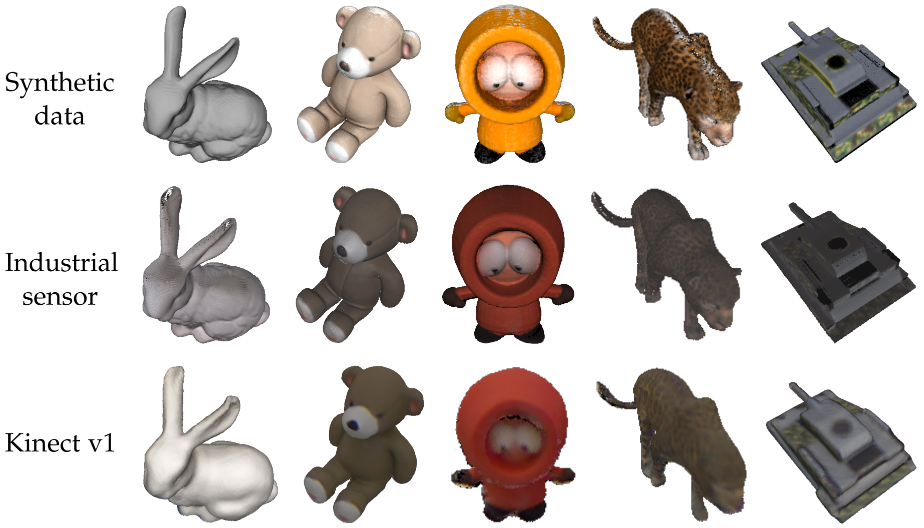


Figure 4.1: **SDF-2-SDF reconstructions of the 3D-printed dataset objects captured with different RGB-D sensors**. Colours vary due to difference between synthetic rendering and 3D-printed models, as well as camera radiometrics.

- introduction of a global pose optimization step, which is elegantly interleaved with the model reconstruction;
- improved convergence via incorporation of photometric and surface orientation constraints;
- the first object reconstruction dataset including ground-truth 3D models, trajectories and RGB-D data from sensors of varying quality.

Our parallel tracking implementation runs in real-time on a multi-core CPU. While pose refinement is only essential when depth data is corrupted by noise, it is interleaved with the final model generation, adding just a few seconds of processing. Furthermore, these two stages can be used as completely standalone tools, and thus can be adopted into any other pipeline.

4.2 Related Work

Fully automatic object reconstruction requires estimation of the precise 6 DoF camera poses from which the RGB-D views were acquired. Arguably, the most widespread strategy for aligning depth data is ICP [15, 37, 180]. While simple and generic, it performs poorly in the presence of gross statistical outliers and large motion. Moreover, it is rather costly due to the required re-assignment of point correspondences in every iteration.

Volumetric registration KinectFusion [102, 157] employs Curless and Levoy's volumetric depth map fusion [45] to represent scene geometry as a continuously incremented SDF, which aids smoothing noise away. Pose estimation is done by rendering the global SDF into a predicted depth image and applying multi-scale point-to-plane ICP for frame-to-model registration. Thus it is susceptible to drift under erratic motion or lack of discriminative geometry. Through a comparison to PCL's implementations of GICP [188] and KinFu [167], we show that SDF-2-SDF can handle cases where both frame-to-frame and frame-to-model ICP variants fail.

Multiple authors [30, 35, 40, 121, 146, 165, 177] report superior registration using implicit surface representations. Notably, Bylow *et al*. [30] and Canelhas *et al*. [35] directly project the points of a tracked frame onto the cumulative SDF in order to avoid the costly correspondence association step of ICP. Similar to Stoyanov *et al*. [206] who leverage point-to-NDT (normal distribution transform) to NDT-to-NDT, we extend the point-to-implicit strategy to an implicit-to-implicit one. Our SDF-2-SDF scheme minimizes the direct voxel-wise difference between a pair of SDFs. Thus it is also correspondence-free, and has further advantages, such as being denser and symmetric, since both SDFs that are being registered steer towards optimal alignment. As a result, it has a wider convergence basin and achieves higher accuracy, as shown by comparisons versus the ROS point-to-implicit implementation of Canelhas *et al*. [33].

Visual odometry DVO is a fast tracking system that works exceptionally well on textured scenes [115, 204]. It employs a photo-consistency constraint to determine the optimal alignment between two RGB-D frames.

Despite requiring a polychromatic support for the object of interest, visual odometry is used in the reconstruction pipelines of Dimashova *et al*. [52] and Kehl *et al*. [111]. These two works then execute a g^{2}o pose graph optimization [122], which undoubtedly yields results of higher geometric fidelity, but is rather computationally demanding. When used in dense scene reconstruction applications, graph-based optimization may last hours to days [257]. We propose improving the estimated trajectory via global implicit-to-implicit optimization. Starting with the SDF model obtained at the end of the online tracking step, the poses of selected keyframes are refined in a frame-to-model fashion. The global SDF is thus iteratively updated and can be readily used as output reconstruction, making our refinement significantly faster.

To once again assess our approach versus both frame-to-frame and frame-to-model techniques, we compare to the authors' implementations of DVO [113] and the method of Kehl *et al*. [111].

Additional constraints Several ICP variants imposing photometric constraints in order to avoid registration failure when geometry is not sufficiently discriminative (RGBD-ICP [86], colour-ICP [104], multi-feature ICP [186]). The previously mentioned object reconstruction pipeline of Kehl *et al*. [111] utilizes colour in both its tracking stage, as part of DVO, and in its final fusion stage after pose optimization has been completed. The refined keyframes are fused using a modification of Zach *et al*. [249]'s TV-L^1 minimization scheme, which takes into account the colour associated with each SDF voxel. Similarly, Bylow *et al*. [29] demonstrate that a voxel grid colour term improves registration accuracy, especially in the absence of rich geometric features. As the SDF-2-SDF formulation allows for straightforward incorporation of additional voxel-wise constraints, we also associate RGB values with voxels.

Another possibility to increase tracking precision is through further geometric terms. Masuda [146] employs the difference between normal vectors to this end. We instead utilize the dot product as a more accurate measure of surface orientation similarity. Although our approach works well without the inclusion of these colour and normal constraints, they are straightforward to integrate and further boost performance.

RGB-D datasets A thorough evaluation of a 3D object reconstruction system requires the availability of both ground-truth trajectories and ground-truth object models. The TUM RGB-D benchmark [208] includes an ample set of sequences with associated poses, while the ICL-NUIM dataset [82] provides the synthetic model of one scene. However, both are designed for SLAM scenarios and therefore feature large spaces rather than smaller-scale objects. Existing RGB-D collections of household items, such as that of Washington University [123], Berkeley's BigBIRD [192] and the texture-less T-LESS dataset [88], either lack noiseless meshes or continuous 6 DoF pose information. Therefore we 3D-printed a selection of objects with different geometries, sizes and textures and scanned them with several RGB-D devices of various quality. Thus we contribute, what is to the best of our knowledge, the first object dataset with original CAD models and RGB-D data from various sensors, acquired from externally measured trajectories.

4.3 SDF-2-SDF

Our object reconstruction pipeline is depicted in Figure 4.2. The object of interest is assumed to be placed on a flat surface, and masked via fast geometric point labelling [111, 183]. Optionally, depth images are de-noised via bilateral filtering [220] or anisotropic diffusion [228]. As opposed to other volumetric methods that require manual volume initialization [30, 35, 157], we automatically estimate the bounding box by back-projection of all masked depth map pixels. Next, the volume is slightly padded and used for the generation of both SDFs that are to be aligned.

These steps are applied to each depth image input to our tracking method, which performs frame-to-frame SDF-2-SDF registration between projective SDFs. We prefer this strategy in order to avoid the error accumulation that frame-to-model approaches are susceptible to, and to allow for a moving volume of interest. At the end of this tracking stage we obtain a weighted average SDF that can be converted to a coloured mesh via marching cubes. However, if a noisy depth sensor has been used, posterior refinement is beneficial. In this case a predefined number of keyframes is globally SDF-2-SDF-registered to their weighted average SDF, circumventing the need for a pose graph. This frame-to-model refinement is applied in a coarse-to-fine scheme with respect to voxel size. Note that the described tracking and optimization stages are entirely stand-alone, and can therefore be combined with other techniques.

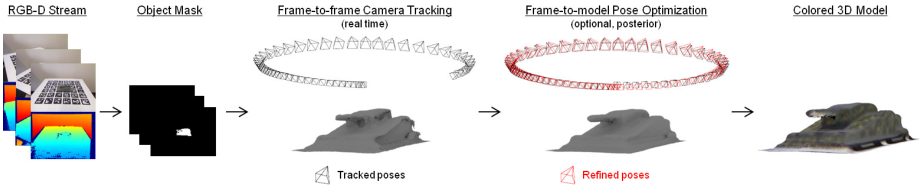


Figure 4.2: **SDF-2-SDF object reconstruction pipeline**: the bounding box of the object is automatically determined for every frame by masking and back-projection, after which it is discretized into voxels. Pairs of frames are then SDF-2-SDF registered. Once this online tracking stage is complete, keyframes are jointly SDF-2-SDF optimized in less than a minute against their weighted average. The system runs entirely on the CPU and outputs a coloured model.

As we are dealing with rigid registration, the unknown Y from Eq. (3.12) is a 6 DoF pose represented via exponential coordinates $\xi \in \mathbb{R}^6$. The same transformation affects all voxels, thus their contributions can be straightforwardly added up into the geometric energy that rigidly aligns two SDFs:

$$E_{geom}(\xi) = \frac{1}{2} \sum_{voxels} \left(\phi_{reference}\omega_{reference} - \phi_{current}(\xi)\omega_{current}(\xi) \right)^2 , \tag{4.1}$$

where $\phi_{reference}$ is the projective SDF of the last frame generated from the identity pose, while $\phi_{current}$ is the projective SDF of the current depth frame generated from its current pose estimate ξ, which is iteratively optimized. The respective weight fields $\omega_{reference}$ and $\omega_{current}$ are used to discard unreliable voxels from the computation. To ease notation, when summing over all voxels we omit the coordinates, *i.e.* we write $\phi_{reference}$ instead of $\phi_{reference}(\mathbf{V})$ in sums.

The intuition behind E_{geom} is that when best alignment between frames is achieved, their per-voxel difference is minimal: truncated voxels have the same values, while the near-surface non-truncated voxels from both grids steer convergence towards surface overlap. Registration is facilitated by the fact that both SDFs encode the distance to the common surface.

The grid structure used in the SDF-2-SDF formulation allows for straightforward incorporation of additional constraints that can be expressed over voxel grids. We propose two terms on the surface voxels, which we approximate as the non-truncated voxels in the narrow band of an SDF grid. In particular, we require overlapping voxels to have the same surface orientation, E_{norm}, and the same colour in each channel, E_{RGB}:

$$E_{norm}(\xi) = \sum_{\substack{surface\\voxels}} \left(1 - \overline{\mathbf{n}}_{reference} \cdot \overline{\mathbf{n}}_{current}(\xi) \right) , \tag{4.2}$$

$$E_{RGB}(\xi) = \frac{1}{6} \sum_{\substack{surface\\voxels}} \sum_{\substack{channel\\j\in\{R,G,B\}}} \left(\zeta^j_{reference} - \zeta^j_{current}(\xi) \right)^2 . \tag{4.3}$$

Note that as we are considering only near-surface voxels, we do not need to additionally use the weight fields to disregard unreliable voxels.

Here it is important to note that since the SDF gradient equals the normals at surface locations, we do not have to store an additional grid. Furthermore, this means that $E_{geom} + E_{norm}$ is a higher-order approximation of the underlying continuous shape than E_{geom} alone. Thus our expectation is that, given data with little to moderate noise, registration will be slightly more accurate and converge faster. On the other hand, we expect E_{RGB} to be helpful in situations with low geometric detail, but richer texture.

The full SDF-2-SDF rigid alignment energy combines all terms, with relative influence determined by the factors $w_{geom} > 0$, $w_{norm} > 0$, $w_{RGB} > 0$:

$$E_{SDF}(\xi) = w_{geom} E_{geom}(\xi) + w_{norm} E_{norm}(\xi) + w_{RGB} E_{RGB}(\xi)\,. \tag{4.4}$$

4.3.1 Camera Tracking

Frame-to-model tracking can be detrimental in object reconstruction, as errors in pose estimation may introduce incorrect geometry when fused into the global model, and consequently adversely affect the subsequent tracking. Therefore, we favour frame-to-frame camera tracking on single-frame projective SDFs.

We determine the relative transformation between two RGB-D frames by setting the pose of the first one to identity and incrementally updating the other one. The tracking minimization scheme for the geometry term is based on a first-order Taylor approximation around the current pose estimate ξ^k in iteration k (Eq. (4.5), (4.6), (4.7)). Similar to other rigid registration approaches, it leads to an inexpensive 6×6 linear system (Eq. (4.8)). Weighting terms have been omitted from formulas for clarity. In order to avoid numerical instability, we take a step of size β towards the optimal solution ξ^* (Eq. 4.9). In each iteration $\phi_{current}$ is generated from the current pose estimate ξ^k, because this strategy yields more accurate values than repeated interpolation. We terminate when the translational update falls below a threshold [208].

$$\mathbf{A} = \sum_{voxels} \nabla_\xi^\top \phi_{current}(\xi^k)\ \nabla_\xi \phi_{current}(\xi^k)\,, \tag{4.5}$$

$$\mathbf{b} = \sum_{voxels} \left(\phi_{reference} - \phi_{current}(\xi^k) + \nabla_\xi \phi_{current}(\xi^k)\ \xi^k \right) \nabla_\xi^\top \phi_{current}(\xi^k)\,, \tag{4.6}$$

$$\frac{\mathrm{d}E_{geom}}{\mathrm{d}\xi} = \mathbf{A}\xi - \mathbf{b}\,, \tag{4.7}$$

$$\xi^* = \mathbf{A}^{-1}\mathbf{b}\,, \tag{4.8}$$

$$\xi^{k+1} = \xi^k + \beta \left(\xi^* - \xi^k \right)\,. \tag{4.9}$$

In the equations above $\nabla_\xi\, \phi$ denotes the Jacobian of the point $\mathbf{V} \in \mathbb{R}^3$, denoting the voxel center, with respect to the pose ξ. It is obtained by the chain rule:

$$\nabla_\xi\, \phi(\mathbf{V}(\xi)) = \nabla_\mathbf{x}\, \phi(\mathbf{V}) \frac{\partial \mathbf{V}}{\partial \xi} = \nabla_\mathbf{x}\, \phi(\mathbf{V})\ \left(\mathbf{I}_{3\times 3} \quad | \quad -[\mathbf{V}(\xi^{-1})]_\times \right)\,, \tag{4.10}$$

where $\mathbf{I}_{3\times3}$ is the 3×3 identity matrix, ξ^{-1} denotes the inverse of the rigid pose represented by exponential coordinates ξ, and the operator $[\cdot]_\times$ returns the skew-symmetric matrix of its argument. Thus $\nabla_\xi \phi \in \mathbb{R}^{1\times 6}$.

Each colour grid channel is a scalar field, so it is treated identically to E_{geom}.

As the normals of the SDF equal its spatial gradient, the surface orientation term imposes curvature constraints, whose derivation is mathematically equivalent to a second-order Taylor approximation of E_{geom}. Thus the objective remains the same, but convergence is speeded up. The derivative of E_{norm} with respect to each component i of the exponential coordinates is:

$$\frac{\mathrm{d}E_{norm}}{\mathrm{d}\xi_i} = \sum_{\substack{surface\\voxels}} -\overline{\mathbf{n}}_{reference} \cdot \left(\nabla_{\mathbf{x}}\, \overline{\mathbf{n}}_{current}(\xi)\ \frac{\partial \mathbf{V}}{\partial \xi}\ \delta_i \right), \tag{4.11}$$

where δ_i is a 6-element one-hot vector of zeros with i-th component 1, and $\nabla_{\mathbf{x}}\overline{\mathbf{n}} \in \mathbb{R}^{3\times3}$ is the spatial gradient of a normal vector, as explained in Section 3.2, which evaluates how the orientation changes with location, *i.e.* it is a measure of curvature.

The complete derivations of all equations presented here are given in Appendix A.

4.3.2 Global Pose Refinement

After tracking, a pre-selected number of regularly spaced keyframes are taken for generation of the final reconstruction. The weighted average scheme of Curless and Levoy [45] provides a convenient way to incorporate the information from all of their viewpoints into a global model ϕ_{model}, which will now act as the reference SDF in Eq. (4.1).

However, when using noisy data the estimated trajectory might have accumulated drift, so the keyframes' poses need to be refined to ensure optimal geometry. For this task we propose a frame-to-model scheme based on the SDF-2-SDF energy. Each pose ξ_t is better aligned with the global weighted average ϕ_{model}. In effect, the optimization is interleaved with the computation of the final reconstruction, and takes less than 30 seconds for 24 keyframes. As we already have good initial pose estimates from the tracking stage, a cheaper gradient descent minimization with step α is sufficient:

$$\frac{\mathrm{d}E_{geom}}{\mathrm{d}\xi} = \sum_{voxels} \left(\phi_{current}\,(\xi) - \phi_{model}\right) \nabla_{\xi}\ \phi_{current}(\xi)\,, \tag{4.12}$$

$$\xi_t^{k+1} = \xi_t^k - \alpha \frac{\mathrm{d}E_{geom}(\xi_t^k)}{\mathrm{d}\xi}\,. \tag{4.13}$$

The pose of the first camera determines the world coordinate frame and is fixed to identity throughout the final optimization. In each iteration, the pose updates of all other keyframes are determined using the global model, after which they are simultaneously applied. To keep the objective fixed, the weighted average is recomputed every couple of iterations, *e.g.* on every $10th$ iteration, rather than on every step. Furthermore, the gradient descent procedure is applied in a coarse-to-fine scheme over the voxel size to ensure that larger pose deviations can also be recovered. As we seek to keep this final post-processing stage as quick as possible, we do not employ the photoconsistency and surface orientation terms. The derivation is also given in Appendix A.

Implementation

The SDF-2-SDF energy is highly parallelizable, because the contributions of each voxel are independent. However, as we estimate the bounding box on the fly, their amount is not known beforehand and varies for each frame. Hence the number of reduction operations which ultimately lead to the 6×6 system of Eq. (4.8) is unknown. In contrast, KinectFusion [157] and point-to-implicit approaches [30, 35] register a VGA-sized depth image either to a point cloud or an SDF, respectively. Thus in these methods there is an upper bound on the number of reduction operations and they can be implemented efficiently on the GPU. In our case this is not guaranteed, so instead, we opt for a parallelized CPU solution on an 8-core Intel i7-4900MQ CPU with 32 GB RAM at 2.80 GHz.

As tracking at a voxel size smaller than the sensor resolution is futile, we used 2 mm, which is the expected error of our noisiest sensor, the Kinect. Our approach involves SDF generation at every iteration after the pose estimate has been updated, so we use SSE instructions to ensure it is done as efficiently as possible. This leaves the computation of each voxel's contribution to the 6×6 system as the bottleneck. To speed it up, we only process voxels with positive weight, and different values in the two grids, achieving real-time performance between 17 and 22 FPS on objects of the scale of household items and toys.

On the other hand, our pose optimization scheme has a simpler mathematical formulation that requires only the calculation of a 6-element vector update in each gradient descent step. We employ a pyramid scheme over voxel size with levels 4 mm, 2 mm, optionally 1 mm. It ensures that initially larger deviations are handled, after which smaller-scale ones are compensated and then used for the final high-resolution model. Typically the whole refinement stage takes less than half a minute, even in the case of severe drift.

4.4 Evaluation

In this section we present exhaustive evaluation on synthetic and real RGB-D input. In all scenarios we assume a single rigid object of interest.

Test Set-up and Datasets

First we consider related methods to compare against, then we discuss the appropriate datasets and metrics over which to evaluate.

Approaches As our tracking and pose optimization routines can be used stand-alone, we evaluate them separately. We denote our tracking-only component as *SDF-2-SDF-reg*, while *SDF-2-SDF* refers to the method with refinement. Unless otherwise specified, we only use E_{geom} for higher speed and lower memory consumption than with the optional constraints E_{norm} and E_{RGB}.

We compare our tracking accuracy to:

- *GICP*: PCL's [167] frame-to-frame generalized ICP [188];
- *KinFu*: PCL's KinectFusion [1, 157] as a frame-to-model ICP variant;

- *FM-pt-SDF*: the frame-to-model point-to-implicit techniques of Bylow *et al.* [30] and Canelhas *et al.* [35], available as a ROS package [33];
- *FF-pt-SDF*: our frame-to-frame modification of FM-pt-SDF [33];
- *DVO-object*: the publicly available implementation [113] of dense visual odometry without refinement [115], applied only over the object, not using its surroundings for registration;
- *DVO-full*: DVO over the entire scene, still without refinement.

Since KinectFusion does not include an explicit optimization step, although frame-to-model registration can be considered as a way of integrating global information, the fidelity of our non-optimized reconstruction was assessed against that of *KinFu*. The refined model was compared to that of Kehl *et al.*'s pipeline [111], which tracks by DVO, detects loop closure, optimizes keyframe poses via g^{2}o [122], and integrates them via TV-L^1 minimization [249] over coloured SDFs, *i.e.* it is a fairly comparable method that also includes posterior refinement.

Datasets Our goal has been to develop a method that is generic with respect to sensor noise characteristics, scanning motion, and object geometry and texture. Moreover, as we are interested in the usability of models, we want to assess not only tracking, but also reconstruction accuracy on real data. Therefore, we use several public datasets, acquired with different sensors.

As already discussed, the availability of benchmarks for object reconstruction is far more limited than for SLAM. We use several examples from the *TUM RGB-D benchmark* [208] and from the *Large Dataset of Object Scans* [38]. The latter provides reconstructions obtained via a robust combination of KinFu's frame-to-model ICP and DVO's RGB-D photometric error. However, they both lack ground-truth 3D models, so we additionally recorded our own *3D-Printed RGB-D Object Dataset*, shown in Figure 1.2(a).

As the name suggests, it contains a selection of 3D-printed objects with diverse geometry, size and colours. Our five objects exhibit various richness of geometry and texture: uniformly coloured (*bunny*), coloured in patches (*teddy, Kenny*), densely coloured (*leopard, tank*); very small (*Kenny*), very large (*teddy*); with thin structures (*leopard*'s tail, *tank*'s gun), with spherical components (*teddy, Kenny*) and symmetries (*teddy, Kenny, tank*). Our intention is that symmetries and elements of poor geometry will be challenging for geometric registration methods, while scarcely textured ones will be difficult for visual odometry-based techniques. These models were 3D-printed in colour with a *3D Systems ZPrinter 650*, which reproduces details of resolution 0.1 mm [251]. Thus we ensure that the textured ground-truth CAD models are at our disposal for evaluation, eliminating any dependence on the precision of a stitching method or system calibration that existing datasets entail.

To capture increasing levels of sensor noise, we used *three RGB-D cameras*: noise-free synthetic rendering in *Blender* [16], an industrial phase shift sensor of resolution 0.13 mm, and a Kinect v1. We recorded in *two scanning modes: turntable and handheld* with the Kinect. We also simulated them in *Blender*

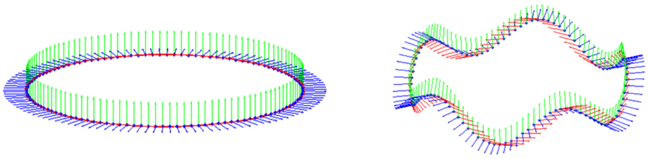


Figure 4.3: **Synthetic trajectories** used for simulating noiseless RGB-D input in *Blender*.

as 120-pose trajectories of radius 50 cm, where the handheld one is a sine wave with frequency 5 and amplitude 15 cm, as shown in Figure 4.3. Thus the synthetic *groundtruth trajectories* are known, while the Kinect poses are obtained from a markerboard placed under the object. The industrial sensor takes 4 seconds to acquire a single RGB-D pair, permitting us to only record turntable sequences. Due to its limited field of view, we could not place a sufficiently large markerboard, so we only use it for evaluation of model accuracy. In all cases the object of interest is placed on a richly textured support that provides optimal conditions for visual odometry, ensuring fair comparisons.

Metrics We take inspiration in the typical RGB-D benchmark metrics [208], but modify them to be more appropriate for object reconstruction. We will use them in their original SLAM formulation in the respective chapter. First, we evaluate the *relative pose error* (RPE) [208] per frame transformation:

$$RPE_{t\to t+1} = (\mathbf{P_i}^{-1}\mathbf{P_{t+1}})^{-1}(\mathbf{Q_t}^{-1}\mathbf{Q_{t+1}}), \tag{4.14}$$

where t is the frame number, $\{\mathbf{Q_{1\dots n}}\}$ is the ground-truth trajectory and $\{\mathbf{P_{1\dots n}}\}$ is the estimated one. It evaluates the difference between the ground-truth and estimated transformations, and equals identity when they are perfectly aligned. In addition, we report the *angular error* per transformation. Both the translational and angular errors are also evaluated for the *absolute* poses. Note that while our relative metric is identical to the RGB-D benchmark RPE per frame, our absolute metric is, in general, more severe than its absolute trajectory error (ATE) [208]. This is because the ATE targets SLAM scenarios and first determines the best alignment between the two trajectories, while in our case they both start with the same initial reference pose, because this directly influences the way frames are fused into models.

Finally, we evaluate the reconstruction error against the original CAD model used for 3D-printing, via the cloud-to-model evaluation of *CloudCompare* [42].

Tracking Accuracy

Synthetic data We start our evaluation with a proof of concept, tested through noise-free synthetic data and summarized in Figure 4.4. SDF-2-SDF-reg clearly outperforms the other methods with an average relative drift below 0.4 mm and angular error below 0.06° for all objects and trajectories. In addition, the maximum errors never surpass 1.55 mm and 0.25° respectively. An analogous trend is observed for the absolute errors: the average translational deviation is

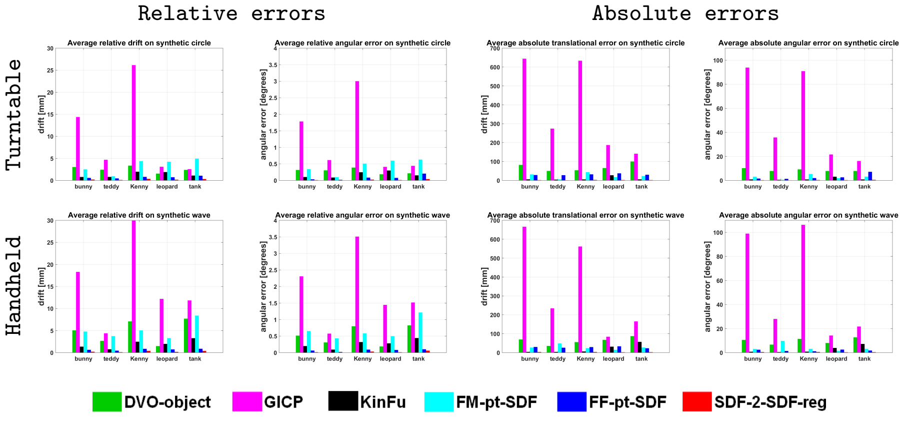


Figure 4.4: **Comparison of tracking errors on synthetic sequences** from the *3D-Printed RGB-D Object Dataset*.

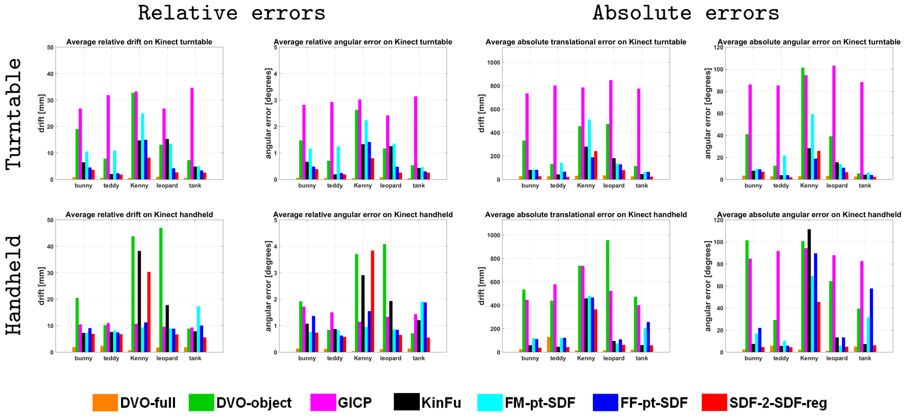


Figure 4.5: **Comparison of tracking errors on Kinect sequences** from the *3D-Printed RGB-D Object Dataset*.

approximately 2 mm, which corresponds to the used voxel size and therefore suggests that given high quality data, only the grid resolution limits our tracking accuracy. The angular deviation always stays below 1°. Notably, our approach performs equally well regardless of object geometry and yields a negligible error with respect to the trajectory size.

FF-pt-SDF and KinFu are closest to our precision on the relative metrics, albeit being at least 2-3 times higher. KinFu exhibits similar behaviour on the absolute metrics, while FF-pt-SDF worsens. This suggests that the relative metrics tend to give advantage to frame-to-frame methods. Nevertheless, SDF-2-SDF-reg is not affected by this bias. As the point clouds are rather small, GICP fails to deliver reliable results and is consistently the worst method for these datasets. Visual odometry and FM-pt-SDF are usually slightly less precise than FF-pt-SDF. The inferior performance of DVO on these small objects indicates that it is designed for scenes with large data clouds rather than small-scale tracking, where it tends to converge locally.

Kinect sequences Figure 4.5 shows that overall the results on the Kinect sequences exhibit similar trends to the synthetic case, although the errors are considerably higher. Typically DVO-full and our SDF-2-SDF-reg are most precise, while GICP and DVO-object are least accurate. The relative drift of SDF-2-SDF-reg ranges between 2 mm on large objects like *teddy* and 10 mm on more challenging ones. The relative angular error is below 1°, and is often almost negligible, *e.g.* 0.19° on turntable *teddy*, and 0.26° on *tank* and *leopard*, which are difficult objects with thin structures. The *Kenny* sequence is a notable exception, since it is composed of a sphere and an ellipsoid, making its back completely symmetric and thus extremely challenging for geometric registration methods. Furthermore, the Kinect is often unable to capture *Kenny*'s arms as they are rather thin, and its high level of noise poses a problem for all methods.

If the richly textured support is taken into consideration for registration, DVO-full outperforms all methods on the relative metrics. However, if only data on the object of interest is used, DVO-object performs much worse. Industrial scenarios often require the inspection of texture-less objects in their working environment, which may not be modified by placing additional support structures. Thus our SDF-2-SDF-reg is designed to be able to function independent of texture, while we would employ colour only if the geometric energy terms entirely fail. It is important to note that despite using only geometric constraints only over the object of interest, we achieve lower absolute errors than DVO-full on *teddy* in both scanning modes, and outperform it on *tank* and *bunny* on turntable trajectories.

All results indicate that SDF-2-SDF-reg is superior to the other volumetric methods. In most cases KinFu is more accurate than both point-to-implicit implementations, of which frame-to-frame tends to be slightly better than the frame-to-model variant. The reason for the poorer performance of these methods is that they register a sparse point cloud to a dense SDF or to another cloud. Therefore, when the object is small there are very few data points, whose measurements are unreliable in the presence of severe noise. Thanks to the inherent smoothing properties of volumetric representations, SDF-2-SDF-reg copes better with such issues. Moreover, it relies on a denser set of correspondences: on average, the used clouds consist of $8 \cdot 10^3$ data points, while the SDFs have $386 \cdot 10^3$ voxels.

A notable failure case for FM-pt-SDF was the turntable *teddy*, where symmetry on the back caused drift from the middle of the sequence onwards, which lead to unrepairable errors in the global model and consequently flawed tracking. Similarly, FM-pt-SDF performed poorly on the turntable *Kenny* due to its fine structures, while FF-pt-SDF did not suffer from error build-up and was most accurate. Clearly, FF-pt-SDF is more suited to camera tracking based on a single object of interest than FM-pt-SDF. In contrast, SLAM scenarios feature large scenes, so areas are often repeatedly scanned and thus provide opportunities for frame-to-model tracking to recover from drift. However, in object scanning every incoming frame usually exposes large unseen areas of the object for which a model of the geometry recovered so far is not as helpful. Due to this reasoning we designed SDF-2-SDF-reg to track in a frame-to-frame fashion.

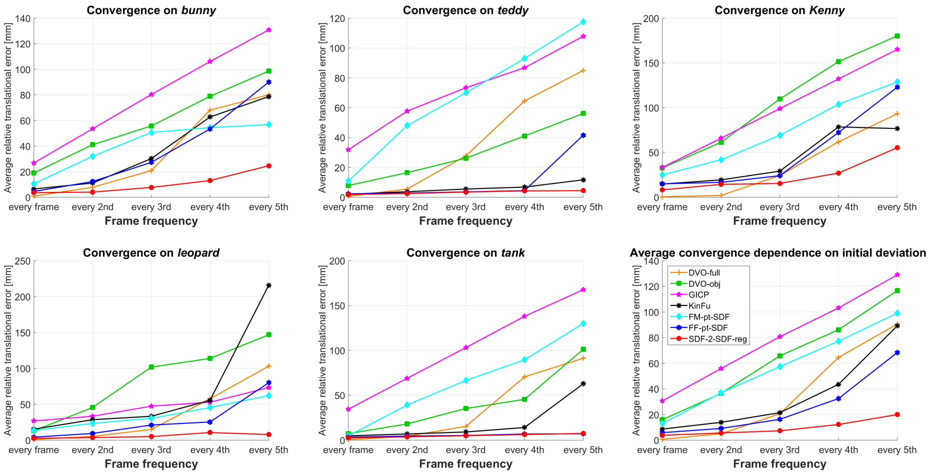


Figure 4.6: **Convergence analysis of registration methods** with respect to frame distance, simulating larger initial deviation on Kinect turntable data.

Convergence Basin

To deepen our analysis of SDF-2-SDF registration, we investigate its convergence basin next. We simulated initial conditions in which the global minimum is gradually further away, by skipping frames from the original turntable Kinect sequences. The first five plots in Figure 4.6 display comparisons to other techniques on each object, while the last plot is averaged over all of them. In the majority of cases, the errors of most methods grow approximately linearly with distance, but SDF-2-SDF-reg has the slowest rate. Thus thanks to its denser formulation and the existence of meaningful values everywhere in the volume, it can determine an accurate pose from a much larger initial deviation of up to approximately 15°.

Notably, with the exception of two-frame distance on *Kenny*, SDF-2-SDF-reg is considerably more precise than DVO-full. The remaining results exhibit a trend similar to what we observed in Figure 4.5: GICP, DVO-object and FM-pt-SDF have the fastest error growth rates, and are outperformed by KinFu and FF-pt-SDF, which behave alike. In particular, the errors of FF-pt-SDF and SDF-2-SDF-reg are nearly identical for each frame distance on the *tank* sequence. This indicates that a frame-to-frame strategy is more advantageous than frame-to-model for a larger pose difference. The reason is that a model is of limited help here, as concluded earlier, because a new frame exposes more unseen parts of the object. On the other hand, the previously observed parts can steer into local minima. Moreover, as the *tank* has a relatively uncomplicated geometry, the point-to-implicit and implicit-to-implicit methods behave similarly. However, the remaining objects, where geometry is more peculiar, present a harder challenge to point-to-implicit strategies, since their point clouds are more susceptible to noise than SDFs. These observations once again confirm our design choices for the SDF-2-SDF framework.

Table 4.1: **Effect of curvature constraints on convergence rate**: comparison of number of iterations to converge when tracking with the signed distance term only (E_{geom}) versus combined with the surface orientation term ($E_{geom} + E_{norm}$).

Object	**Iterations to convergence**			
	Turntable		Handheld	
	E_{geom}	$E_{geom} + E_{norm}$	E_{geom}	$E_{geom} + E_{norm}$
bunny	42.09	23.37	41.26	31.18
teddy	17.28	15.60	25.16	17.22
Kenny	46.01	28.36	76.44	41.09
leopard	24.03	17.29	34.52	22.49
tank	28.88	19.73	41.86	29.02

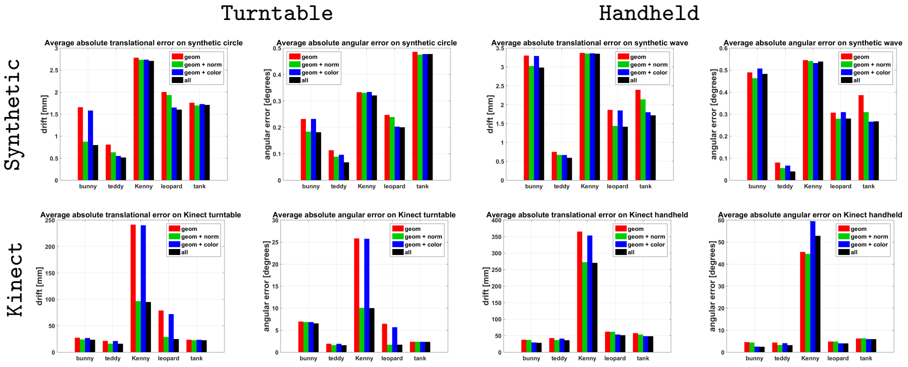


Figure 4.7: **Effect of surface orientation and colour constraints** on the average absolute error of SDF-2-SDF-reg tracking.

Contributions of Additional Constraints

Next, we evaluate the effect of surface orientation and photoconsistency on the registration error, summarized in Figure 4.7. We obtained similar results with weight values from the set $\{0.05, 0.1, 0.2\}$ for both w_{norm} and w_{RGB}. While there is no considerable change on the synthetically rendered RGB-D sequences, indicating that high quality depth data is sufficient for highly accurate registration, on Kinect data each additional constraint decreases the error of E_{geom} by a certain amount. This depends on the properties of the object, while all three constraints together make E_{SDF} most accurate. An exception is again the angular error of the challenging handheld *Kenny* dataset. Its error decreases with the normal term, but increases with the texture one. We suppose that this is due to imprecise depth-to-color camera calibration, which can lead to a significant offset on a small object like this.

Considering that the additional terms entail more calculations, we advocate to use them depending on the specific case. For instance, E_{norm} is very beneficial on the *teddy*, since it is a large object, where normals can be estimated reliably. Colour helps on richly textured objects, like *leopard* and *tank*. Finally, it is noteworthy that in several cases the final error of E_{SDF} has become even lower than that of DVO-full from Figure 4.5, where it previously was not.

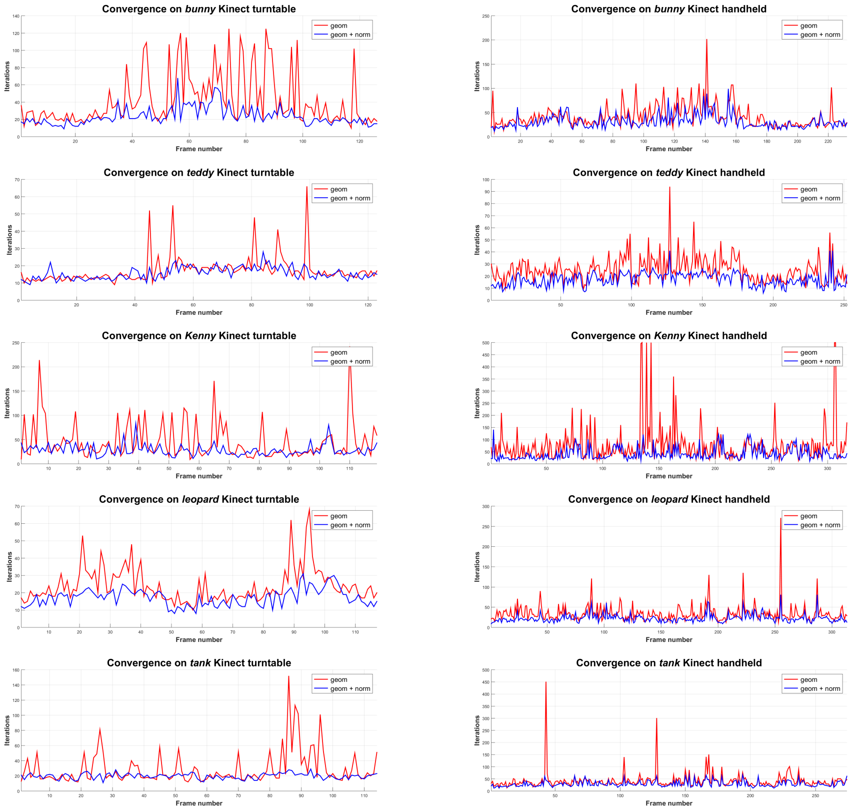


Figure 4.8: **Comparisons of iterations until convergence** with E_{geom} versus $E_{geom} + E_{norm}$ on Kinect sequences.

As previously discussed, since surface normals equal the normalized gradient of the signed distance field, E_{norm} is a second-order term in addition to E_{geom}, which does not significantly change the optimum of the energy. However, assuming that the computation of normals is not heavily influenced by noise, we expect that this optimum will be reached in fewer iterations. We investigate this claim on noisy Kinect data in Table 4.1 and Figure 4.8.

First, we notice that handheld sequences typically require more iterations to reach convergence than turntable ones. This is due to the more erratic scanning motion, which causes the effects of motion blur and rolling shutter to be more noticeable. In addition, bigger objects, like *teddy*, require less iterations than smaller ones, like *Kenny*, since they contain more data, a smaller proportion of which is influenced by noise. Last but not least, the expectation for less iterations with the normal term is confirmed in all cases. The plots indicate that E_{norm} remedies cases when E_{geom} alone did not converge, since spikes in the red curves are not present in the blue ones. There are rare cases in which the combined energy needs several more iterations than the geometric one alone, occurring on frame pairs with smaller overlap. As the standard deviation of the required number of iterations decreases noticeably, we conclude that the second-order term regularizes the energy.

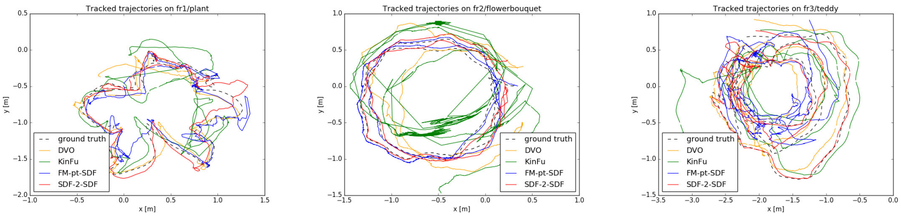


Figure 4.9: **Comparison of estimated trajectories on the *TUM RGB-D benchmark*** [208].

Note that synthetic renderings and high quality industrial sensor sequences typically require less iterations than Kinect ones, even based on E_{geom} only. Therefore, while on Kinect data E_{norm} might lead to convergence in half the iterations, its contribution is not as significant on less noisy data.

Other Public RGB-D Datasets

As a last part of our evaluation of tracking performance, we test on sequences from the *3D Object Reconstruction* category of the *TUM RGB-D benchmark* [208]. As the dataset itself is SLAM-oriented, the sequences contain larger-scale moderately cluttered scenes, unconstrained camera motion and occasionally missing depth data due to close proximity to the sensor. Due to this intended use we only test DVO-full, KinFu and FM-pt-SDF, in addition to our approach, as shown in Figure 4.9. The entire images were used for DVO, while all other methods tracked solely using the bounding volume of the object of interest. Nevertheless, our SDF-2-SDF-reg was most precise on *fr1/plant* and *fr3/teddy*, and was only slightly less accurate than FM-pt-SDF on *fr2/flowerbouquet*. The reason is that its leaves have no effective thickness, therefore the SDFs lose their power in discerning inside from outside and, depending on parameters, might oversmooth and become inferior to point cloud registration. This effect can be mitigated by a finer voxel size, but that would entail a longer processing time.

Pose Refinement and Reconstruction Accuracy

We now shift our focus to evaluating the output model accuracy after pose refinement. The results of SDF-2-SDF on our *3D-Printed RGB-D Object Dataset* are displayed in Figure 4.1. While the shapes are reconstructed well, the difference in device quality is apparent. The models obtained from phase shift data are very detailed and synthetic-alike, while those from Kinect are smoothed out. This is most clearly visible on the edges of the *tank*, on the ears of the *leopard* that have not been captured by the Kinect, and from the lack of details on the *bunny* body.

Furthermore, in Figures 4.10 and 4.11 we provide qualitative comparison on both the industrial and Kinect turntable sequences of our *3D-Printed RGB-D Object Dataset* between KinFu, the DVO-full based method of Kehl *et al.*, and SDF-2-SDF without and with refinement. These snapshots reflect the numerical

Table 4.2: ***CloudCompare* evaluation of the absolute cloud-to-model reconstruction error** on the *3D-Printed RGB-D Object Dataset*. SDF-2-SDF-reg refers to our method without refinement, while SDF-2-SDF includes refinement. The variants of Kehl *et al.*'s pipeline [111] indicate whether DVO-object or DVO-full was used for tracking.

Object	**Method**	**Error [mm]**					
		industr. turntab.		Kinect turntable		Kinect handheld	
		mean	std.dev.	mean	std.dev.	mean	std.dev.
bunny	KinFu	0.664	0.654	3.800	2.840	4.101	3.716
	SDF-2-SDF-reg	0.656	0.438	2.586	1.869	1.770	1.733
	Kehl-object	2.149	2.869	5.156	4.115	8.274	6.013
	Kehl-full	0.838	0.860	1.134	1.243	1.124	1.095
	SDF-2-SDF	**0.541**	**0.436**	**0.953**	**0.843**	**0.996**	**0.853**
teddy	KinFu	0.998	0.807	1.271	1.045	2.355	1.447
	SDF-2-SDF-reg	0.930	0.588	1.078	0.890	1.589	1.537
	Kehl-object	1.028	0.892	2.306	1.862	2.287	1.826
	Kehl-full	4.828	4.215	1.221	0.858	3.066	2.380
	SDF-2-SDF	**0.910**	**0.584**	**0.722**	**0.542**	**0.990**	**0.841**
Kenny	KinFu	1.650	1.451	1.511	1.387	2.874	2.727
	SDF-2-SDF-reg	0.363	0.391	1.295	1.311	2.415	2.051
	Kehl-object	1.816	1.710	3.181	3.238	*failed*	*failed*
	Kehl-full	2.553	2.644	**1.263**	**0.850**	**2.282**	**1.381**
	SDF-2-SDF	**0.315**	**0.336**	1.276	1.128	2.358	1.960
leopard	KinFu	1.785	1.299	4.445	2.430	1.886	3.292
	SDF-2-SDF-reg	0.760	0.830	2.692	1.882	1.321	1.220
	Kehl-object	1.018	1.378	5.693	5.050	*failed*	*failed*
	Kehl-full	3.626	3.705	1.907	1.218	1.281	1.218
	SDF-2-SDF	**0.652**	**0.614**	**1.308**	**1.154**	**1.263**	**1.111**
tank	KinFu	1.390	1.315	1.561	1.453	2.579	2.265
	SDF-2-SDF-reg	0.953	0.740	1.336	1.188	2.042	2.404
	Kehl-object	1.573	2.250	1.192	1.009	2.340	2.062
	Kehl-full	2.617	2.571	1.064	0.872	**0.946**	**0.806**
	SDF-2-SDF	**0.466**	**0.416**	**0.911**	**0.745**	1.508	1.760

reconstruction errors, listed in Table 4.2, where we additionally test Kehl *et al.*'s pipeline in its less accurate version based on DVO-object for tracking.

In most cases, SDF-2-SDF-reg yields better results than KinFu even without refinement. In particular, optimization is typically not needed when using phase shift data. On the other hand, it is vital on the more challenging *Kenny*, *leopard* and *bunny* Kinect scans. SDF-2-SDF's error is clearly below 1 mm on all phase shift sequences, and stays below 2 mm on the Kinect ones. As these values correspond to the device uncertainties, we once again confirm that our approach is only limited by the sensor resolution and the voxel size.

Contrary to expectations, the table shows better results for Kehl-object than Kehl-full on industrial data. This is, however, because the provided implementation required resizing the original 2040 × 1080 images to VGA resolution, leading to increased error when processing areas near the image border, where the textured table is located. The results of KinFu and SDF-2-

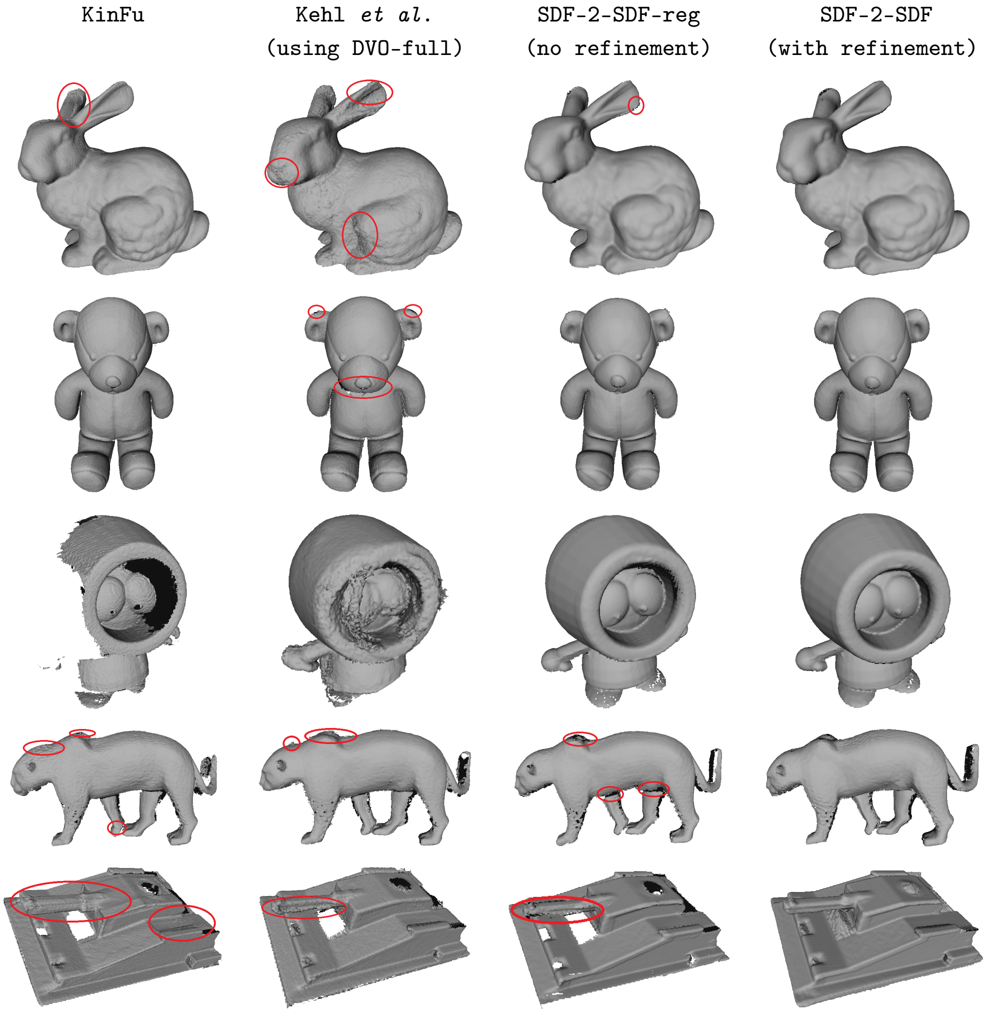

KinFu
Kehl *et al.* (using DVO-full)
SDF-2-SDF-reg (no refinement)
SDF-2-SDF (with refinement)

Figure 4.10: **Qualitative comparison of untextured reconstructions from scans with the high quality industrial sensor**. Object poses might differ slightly, since models yielded by different methods are non-identical. Fine structures cause related approaches to fail, *e.g.* on *Kenny*, or to exhibit misalignment errors, *e.g.* on *bunny*'s ears, *tank*'s gun, connection of *leopard*'s halves.

SDF did not change for VGA and the original size, indicating that volumetric approaches are less sensitive to such issues. Moreover, the speed of SDF-2-SDF remained unaffected, as it only depends on the voxel resolution, and not on the image or point cloud size, while the other methods slowed down with larger image dimensions. Thus our system generalizes well not only for various object geometry, but also for any device.

Both qualitative comparison figures suggest that the large *teddy* is the easiest object for all methods, while the tiny *Kenny* is most difficult, since it is not only more affected by noise, but also uniformly textured and with a symmetric back.

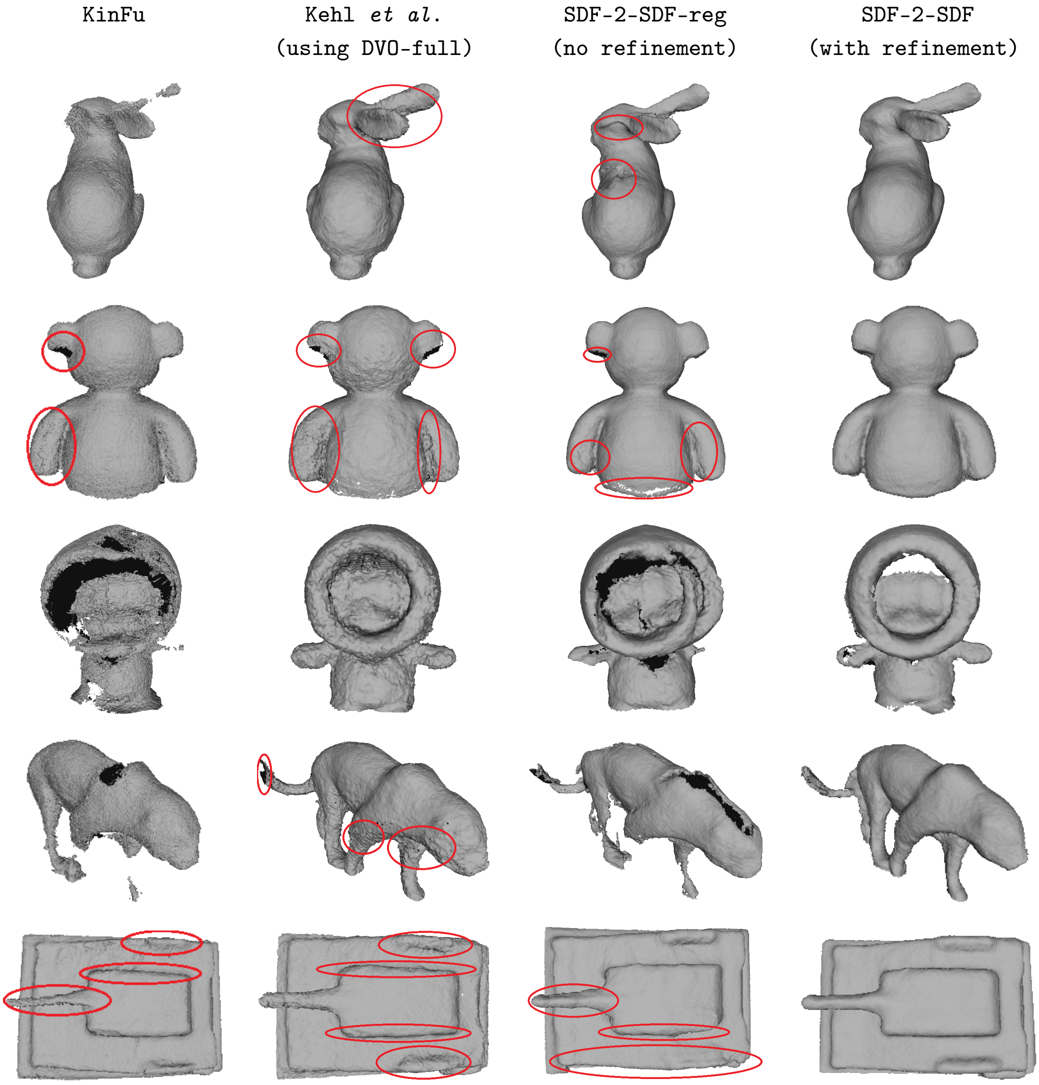

KinFu
Kehl et al. (using DVO-full)
SDF-2-SDF-reg (no refinement)
SDF-2-SDF (with refinement)

Figure 4.11: **Qualitative comparison of untextured reconstructions from Kinect scans** of the objects from our *3D-Printed RGB-D Object Dataset*.

On industrial data SDF-2-SDF-reg produces slightly better reconstructions than Kehl *et al.*, while on Kinect data Kehl *et al.* is superior. However, the model errors indicate that if its DVO tracking component is constrained only to the object, performance becomes significantly worse on Kinect data and might even fail on the more erratic handheld trajectories.

To sum up, our registration technique alone outperforms related methods on high quality depth data, while it requires the posterior refinement step on noisier input, with which it again manages to deliver a highly accurate 3D reconstruction.

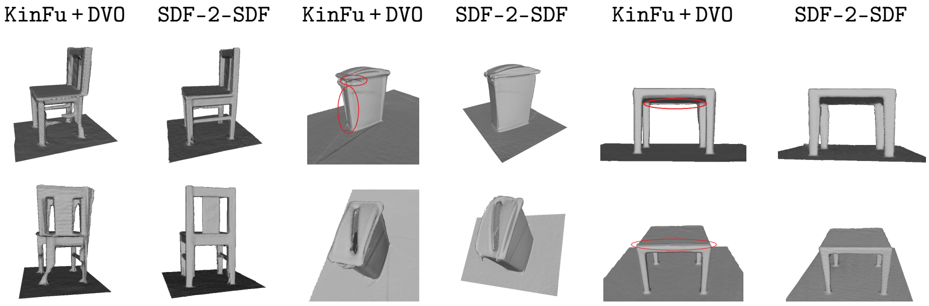


Figure 4.12: **Qualitative comparison versus reconstructions in the *Large Dataset of Object Scans*** [38].

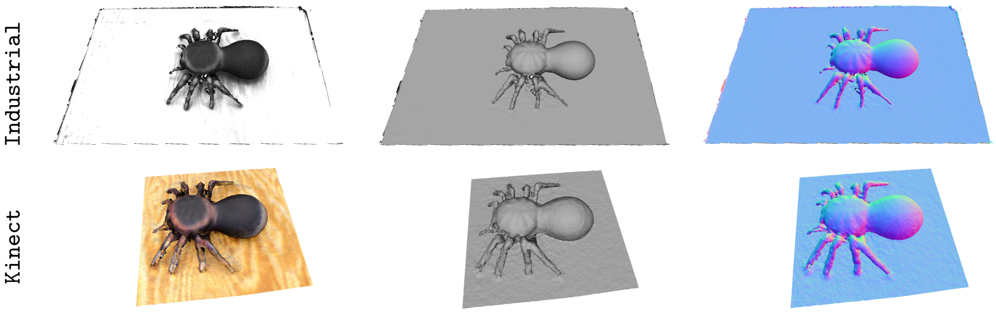


Figure 4.13: **Comparison of sensor quality and SDF-2-SDF reconstructions** on a challenging spider object scanned with an industrial sensor and a Kinect.

Further Qualitative Results

We now present some additional qualitative results. First, to demonstrate the generality of our approach, we test it on bigger objects from the *Large Dataset of Object Scans* [38], and compare to the reconstructions provided by the authors in Figure 4.12. Note that in order to stay within real-time constraints, we used a voxel size of 8 mm for tracking. Even though the dataset reconstructions are obtained via a combination of ICP's geometric error with the photoconsistency of DVO-full, SDF-2-SDF manages to better recover challenging details such as chair legs and support beams over long sequences with thousands of frames.

In addition, Figure 4.13 shows two of our own scans of a spider toy object, acquired with a monochromatic 10 FPS version of the phase shift sensor used for the dataset, and with a Kinect. This is a difficult object because it has very thin legs, which are hard to capture with a sensor, and which are then challenging to accurately register. Nevertheless, SDF-2-SDF manages to reconstruct a geometrically consistent model regardless of the sensor. What is interesting to note here is the texture-less rendering of the reconstructions, as they are very indicative of the difference of quality of the two sensors, even when the reconstruction is accurate.

Table 4.3: **SDF-2-SDF-reg camera tracking module runtime statistics** on the *3D-Printed RGB-D Object Dataset*: average/ fastest/ slowest.

Tracking [**milliseconds** per frame]		
Pre-processing	Reference SDF generation	Minimization iterations
1.7/ 1.6/ 1.8	2.6/ 1.7/ 3.8	45.3/ 41.4/ 54.9
Overall: 49.6/ 44.7/ 60.5 ms ↔ 20/ 22/ 17 FPS		

Table 4.4: **SDF-2-SDF pose optimization module runtime statistics** on the *3D-Printed RGB-D Object Dataset*: average/ fastest/ slowest.

Refinement [total **seconds**]		
Weighted averaging	Optimizing poses	Marching cubes
1.9/ 0.4/ 6.8	6.1/ 0.3/ 20.2	0.6/ 0.2/ 1.3
Overall: 8.6/ 0.9/ 28.3 s		

Runtime

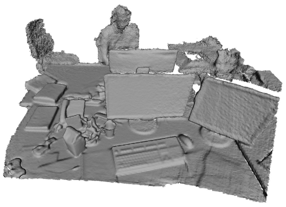
As explained in the method, thanks to SSE instructions and 8-core parallel processing over the narrow band, SDF-2-SDF-reg tracking runs at 17-22 FPS on household objects when a voxel size of 2 mm is used. Tables 4.3 and 4.4 list the time taken for each major step of our pipeline as average over all sequences, as well as the fastest and slowest runs. The quickest processing time was achieved on the small and noise-free *synthetic Kenny*, while the most time consuming was the *Kinect leopard*, since it is a large but thin object and is thus very susceptible to sensor noise, leading to more iterations required to converge.

Our refinement procedure needs at most 40 iterations on each voxel resolution level, taking up to 30 seconds to deliver the output reconstruction, which is generated via marching cubes from the final field. In comparison, Kehl *et al.*'s pose graph optimization took 196.4 s on average (minimum 53 s, maximum 902 s) for the same amount of keyframes. Table 4.4 shows that our pose-graph-free refinement is considerably faster.

4.5 Conclusion

We have developed a complete 3D object reconstruction pipeline that starts with raw sensor data and delivers a highly precise 3D model without any user interaction. The underlying novel implicit-to-implicit registration method is dense and direct, whereby it makes use of all available depth data, while avoiding explicit correspondence search. The proposed global refinement technique is an elegant and inexpensive way to jointly optimize the poses of several views and the reconstructed model. Experimental evaluation has shown that our reconstructions are of higher quality than those produced by related state-of-the-art systems. Last but not least, in contrast to other direct volumetric methods, we achieve real-time tracking on the CPU, which might be critical for certain applications.

Despite the achieved processing at interactive rates for small- to medium-scale objects, it is desirable to extend our method to larger scenes, such as office spaces, *i.e.* to make it applicable to SLAM scenarios. This is challenging since we employ a memory-intensive regular voxel grid and utilize the signs of values in empty space, so a hierarchical memory-efficient data structure is not directly applicable in our case. In addition, the reduction operations for building a 6×6 from an amount of voxels larger than the number of pixels in a VGA image prevent a straightforward GPU implementation. Storing only the signed distances for 512^3 voxels requires 0.5 GB, and for 1024^3 - 4 GB. The storage of voxel weight and colour further increases the memory consumption. Clearly, the problem soon becomes intractable, as processing a high amount of voxels also naturally entails increased runtime. Therefore in the next chapter we investigate other strategies for extending our technique to larger volumes.



5 SDF-based Scene Reconstruction

The next application that we address with our implicit-to-implicit scheme is the 3D reconstruction of larger-scale static objects and scenes, such as rooms and big industrial machines, from depth data. This task is closely related to Simultaneous Localization and Mapping (SLAM) [10, 47, 53, 126], where the goal is to estimate the ego-motion of the camera or robot via techniques such as visual odometry [162] and fuse the observed data into a map, or reconstruction, of the explored environment. More importantly, this has to happen at the frame rate of the used sensor without any latency, so that, for instance, a robot can react to its surroundings in a timely manner.

Here we extend our SDF-2-SDF object reconstruction approach to the real-time SLAM scenario. In this sense, we benefit from the accuracy of the developed implicit-to-implicit registration scheme, but adapt it in several ways to make it suitable for larger volumes and inside-out scanning motion.

First, if the entire observed volume is represented as a regular voxel grid during tracking, it will be extremely slow. Therefore we choose to carry out registration only over the most geometrically discriminative regions of the scene, which are characterized by high curvature. We anchor so-called *limited-extent volumes* (LEVs), which are $8 \times 8 \times 8$ sub-volumes representing partial SDFs, at such salient locations and carry out SDF-2-SDF registration over them.

As the memory consumption is fixed, we are able to efficiently parallelize the pose estimation process and port it to the GPU. Moreover, as the CPU would now be left idle, we instead dedicate it for concurrent joint refinement of the most recently tracked poses. In this way we create a fully online hybrid GPU/CPU method that carries out both tracking and optimization, thus reducing drift without posterior processing.

We use public datasets of both large objects and scenes to quantitatively demonstrate the accuracy of our trajectory estimation, which is particularly advantageous on rotational motion, and the fidelity of our reconstructions.

5.1 Introduction

Simultaneous localization and mapping in real time is among the most pivotal computer vision tasks, with many commercial applications ranging from robotic navigation and scene reconstruction to augmented and virtual reality. Equipped with a hand-held camera, the goal is to explore a static environment, simultaneously determining the 6 degrees-of-freedom camera pose at every time instance and reconstructing the surroundings.

Here we consider the scanning scenario that employs an RGB-D sensor, which eliminates the inherent scale ambiguity that monocular approaches are subject to. The earliest works [61, 86] relied on hand-crafted sparse visual features to match 3D locations via ICP variants [15, 37]. Soon after, the seminal KinectFusion system [157] demonstrated the advantages of volumetric registration through the use of a continuously incremented truncated signed distance field that represents the estimated scene geometry. It was then followed by various extensions that proposed improvements to different aspects of the pipeline, *e.g.* to make the registration energy more robust [30, 35, 109], or to tackle the memory limitations of regular voxel grids, such as moving volumes [176, 237, 240], octrees [203, 205, 253] and voxel hashing [161].

The cumulative SDF globally incorporates information from multiple views and thus the frame-to-growing-model registration scheme can be considered as including a form of global optimization. However, it only allows for drift reduction, without a possibility to reposition incorrectly fused geometry. Most existing approaches that explicitly perform optimization require all depth maps [46, 257] or meshed scene fragments [39, 68, 85, 258] to be stored and lead to lengthy posterior refinement.

This problem is addressed by Parallel Tracking and Mapping (PTAM) [117], which is one of the most acclaimed real-time monocular SLAM techniques. It combines tracking in one thread with global map refinement in another one. The framework that we propose is inspired by this parallel approach, but targets an RGB-D setting. Our key idea is to enable concurrent execution by unifying the efficiency of sparse interest point alignment with the accuracy of dense volumetric approaches, which we observed in our SDF-2-SDF object reconstruction method presented in the previous chapter.

Recall that it utilizes implicit-to-implicit SDF grid alignment to achieve more precise motion estimation than KinectFusion and point-to-implicit approaches [30, 35]. While these techniques register over an amount of data equal to the depth map resolution, SDF-2-SDF processes all voxels. In addition to the associated high memory requirements, the large amount of atomic reduction operations prevents efficient GPU parallelization, thus restricting operation to small spaces that are insufficient for SLAM. Guided by the intuition that geometry-poor locations, such as flat walls, impede registration, we propose to select a fixed number of the most geometry-rich locations in a range image, and anchor small SDF volumes of fixed size around them. Thus only informative data is used for registration, achieving the accuracy of fully dense techniques at a fraction of the cost. Furthermore, this strategy is more straightforward to implement than moving volumes, octrees and voxel hashing. It enables us to

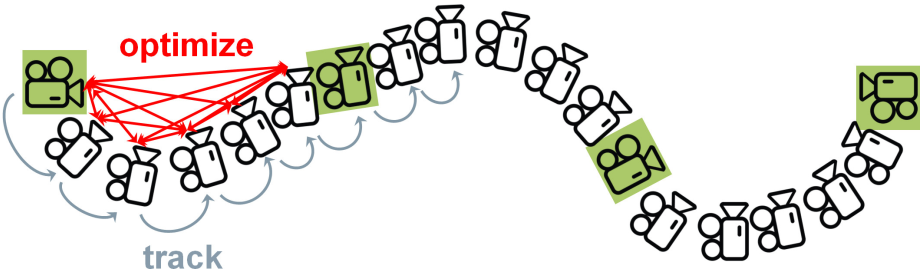


Figure 5.1: **Illustration of our hybrid GPU/CPU concurrent tracking and refinement approach**: the GPU tracking module continues its operation, while the CPU refinement module jointly optimizes the last available batch of tracked frames. Once the next batch is complete, *i.e.* when the third camera highlighted in green has been tracked, the refinement module will switch to the middle batch, while the tracking module will continue independently. To reduce drift even further, we additionally make sure the batches overlap so that the pose of each frame is refined twice.

apply *SDF-2-SDF registration in parallel over all volumes* on the GPU, seeking a common rigid-body motion. Moreover, as the CPU would now be only responsible for the data flow, it is available to perform *concurrent pose refinement* over several of the already registered frames. In this way we create a system that minimizes drift in real-time without the need for posterior global optimization.

To sum up, we propose *SDF tracking and refinement* (SDF-TAR): a real-time system for parallel tracking and refinement based on direct registration between multiple limited-extent SDFs, whose operational principle is sketched in Figure 5.1. Our contributions are the following:

- a novel approach to reduce the memory footprint of volumetric registration, while preserving its accuracy;
- a fully real-time volumetric SLAM method which combines GPU tracking with concurrent CPU pose refinement on overlapping batches of RGB-D frames for online drift reduction.

Quantitative experimental evaluation demonstrates that our *limited-extent volume* (LEV) strategy leads to more precise tracking than related state-of-the-art techniques when the dominant motion is rotational, and on-par results in general settings. In addition, we assess the drift reduction achieved by our batch refinement, which is manifested through higher-fidelity reconstructions.

5.2 Related Work

Here we describe related approaches for scene reconstruction from RGB-D data, with special focus on works that were not mentioned or thoroughly discussed in the object reconstruction chapter.

Volumetric reconstruction KinectFusion [102, 157] and point-to-implicit approaches [30, 35] have already been analyzed at length in this thesis. While they tend to be susceptible to errors under erratic motion and lack of discriminative geometry, our implicit-to-implicit registration scheme leverages both SDFs for more accurate direct alignment. In the previous chapter we used it both for the frame-to-frame tracking and for the subsequent global optimization, obtaining improved trajectory and reconstruction precision in the context of object scanning.

Here we propose to make it suitable for SLAM scenarios as well. We seek to apply it in a fully online fashion following the example of PTAM [117] that executes concurrent tracking and refinement. To this end, the camera is tracked in real-time on the GPU, while a fixed number of already tracked frames are jointly refined on the CPU. As there is no real-time constraint on the refinement, it runs for as much time as the tracking module permits, *i.e.* until enough frames are tracked for the next batch to start being optimized.

Memory load reduction A major limitation of regular voxel grids is their high memory requirement, which limits the operational volume to medium-scale spaces. It has been tackled in various ways, including moving volumes [176, 237, 240], octrees [**?**, 203, 205, 253], voxel hashing [109, 161], non-hierarchical [160] and hybrid hierarchical structures [36]. However, they are beneficial for storing or updating values, but may not as efficient when an SDF needs to be re-generated multiple times per second, as done in our SDF-2-SDF approach when a camera pose is re-estimated.

Moreover, methods that rely on dense image alignment need robust techniques to disregard outliers [62, 115]. On the other hand, approaches like RGB-D SLAM [61] that detect 2D features and match them in 3D, discard a lot of useful information and require RANSAC [69] and pose graph optimization [122] to estimate consistent trajectories. While many authors have addressed 3D keypoint detection [41, 76, 99, 105, 202, 221], the occlusions and noise inherent to consumer-grade RGB-D cameras currently limit their applications to object detection, recognition and classification [5, 18, 57].

Inspired by the accuracy of SDF-2-SDF registration, we aim to apply it to larger-sized objects and SLAM. To this end, we propose a quasi-dense technique which combines the efficiency of keypoint-based methods with the accuracy of dense schemes. First, in order to make GPU parallelization suitable, we make sure that there is an upper bound on the number of voxels that will be used for registration. More precisely, we carry out registration over a fixed number of limited-extent volumes (LEVs), which are small SDFs with fixed side length. We anchor them at locations of distinct geometry and determine a common rigid-body motion from all of them. The volumes capture local geometry and thus grant flexibility with respect to their exact positions. The anchor points are chosen as the locations with highest mean curvature, which is the second-order derivative taken directly over the depth map [89], further facilitating real-time performance. Therefore this strategy ensures not only fixed memory requirements and suitability for GPU implementation, but also accuracy similar to that of full volume SDF-2-SDF.

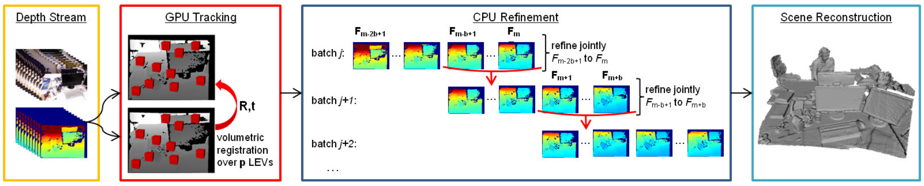


Figure 5.2: **SDF-TAR pipeline**: the relative motion between every two depth frames is estimated on the GPU from p limited-extent volumes, anchored at locations of high curvature. As soon as frame F_m is tracked, the CPU refinement module starts jointly optimizing frames F_{m-2b+1} to F_m. In the meantime tracking resumes on frames F_{m+1} to F_{m+b}. Once this new batch is ready, refinement is switched to frames F_{m-b+1} to F_{m+b}. This strategy ensures highest geometric consistency by optimizing every pose twice.

Global optimization Although refinement can be highly beneficial, it is often not viable for volumetric methods. Due to the high processing requirements of dense data, most existing pipelines resort to expensive posterior optimization that can take hours [39, 68, 85, 257, 258]. This added runtime can be avoided by running refinement concurrently to tracking, as in PTAM [117]. Our pose optimization is also applicable online, as it is done over limited-extent volumes.

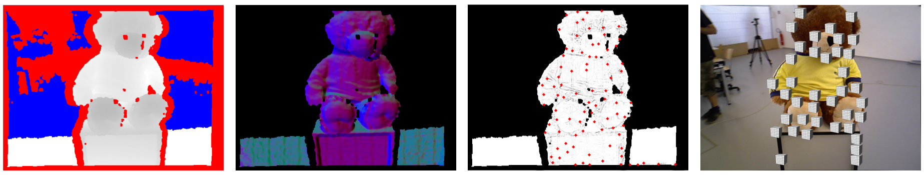
Approaches that include refinement perform it either jointly over all frames, or over a fixed amount of those that were last tracked. For instance, Pirker *et al*. [168] carry out sliding window bundle adjustment, but use sparse 2D-3D correspondences that entail loop closure detection and posterior pose graph optimization. Whelan *et al*. [238] combine incremental as-rigid-as-possible space deformation and every-frame map correction, but depend on the presence of loop closure and add some minimal time latency as more frames are processed. Similarly, ElasticFusion [239, 241] relies on local loop closures to activate non-rigid model-to-model refinement, without further improving the estimated trajectory. Therefore, we identify SDF-TAR as the first pose-graph- and loop-closure-free volumetric RGB-D SLAM method that carries out camera tracking and batch optimization in a fully online fashion.

5.3 SDF-TAR

Figure 5.2 presents the pipeline of our concurrent SDF-based tracking and refinement approach. Next, we describe our limited-extent volume scheme for reducing the memory requirements of regular voxel grid registration. Then we explain how our implicit-to-implicit energy is applied over these partial volumes both for 6 DoF frame-to-frame tracking and for joint refinement, and how we combine these stages into an online hybrid GPU/CPU SLAM system.

5.3.1 Limited-Extent Volumes

Our strategy for reducing the memory load is an easy to implement solution based on the intuition that regions of indiscriminative geometry are not useful



(a) Masked depth. (b) Normal map. (c) Curvature peaks. (d) LEVs.

Figure 5.3: **Limited-extent volume anchor point selection process**: (a) Far-away (blue) and near-edge (red) points are masked out from the depth map to discard potentially noisy values. (b) Normals are calculated as derivatives over depth. (c) Curvature is calculated as derivatives of normals. Its size is non-maximum suppressed to determine peaks separated by a minimal distance. (d) The p highest peaks are used as anchor points for the LEVs, which are small SDFs of fixed size.

for registration may even impede it. On the contrary, geometry-rich locations are highly distinct from their surroundings and can therefore quickly steer registration to an optimal solution.

Thus our key idea is to set p partial SDFs $\Omega_1, ..., \Omega_p$ of resolution $x \times y \times z$ voxels with side length l around the points of highest curvature in the scene. Then we carry out our SDF-2-SDF registration in order to determine a common rigid-body motion ξ for all of these limited-extent volumes (LEVs) simultaneously. This approach guarantees that the memory load will be kept constant for every pair of frames, and thus gives an upper bound for the processing time. Hence we can set all parameters appropriately, so that we will always stay within real-time constraints.

We anchor the LEVs at the points of highest curvature, which are not only discriminative, but also inexpensive to detect, since they can be found via operations over the depth image only, without need to search in 3D. The process is illustrated in Figure 5.3. First, we pre-process the depth image. Since the sensor error increases quadratically with distance [116], we consider measurements further than 2 m as unreliable and discard them. Furthermore, RGB-D cameras tend to be inaccurate near depth discontinuities, thus we also mask out pixels near edges. Next, we estimate the surface normals as derivatives over the preprocessed depth map, following the method of Holzer *et al.* [89]. Then we calculate the curvature magnitude from the derivatives of the normal map. Finally, we apply non-maximum suppression [153], so that only one high curvature point is selected within every window of size $w \times w$ pixels. This ensures that neighbouring LEVs will not overlap. Finally, we select the p points with highest curvature values in the non-maximum-suppressed image, back-project them to 3D and generate partial SDFs of $x \times y \times z$ around them. If there are less than p peaks, we simply take all of them.

5.3.2 Parallel Tracking and Refinement

As we are still dealing with a rigid registration problem, the energy remains the same as in Eq. (4.1). Note that in order to keep processing within real-time constraints, we only use the geometric SDF energy term and do not employ the additional photoconsistency and surface orientation terms. As the energy is now a sum over partial volumes, all of which are influenced by the same transformation ξ, we re-write it as:

$$E_{geom\ LEV}(\xi) = \sum_{\substack{volume\ \Omega_i \\ i=1..p}} \left(\sum_{voxels\ \in\ \Omega_i} \left(\phi_{reference}\omega_{reference} - \phi_{current}(\xi)\omega_{current}(\xi)\right)^2 \right) . \tag{5.1}$$

Note that the locations of the LEVs are determined only over the reference frame, after which their physical volumes are used for the generation of the p partial SDFs of both the reference and the other frame.

Limited-Extent Volume Registration

Following the same reasoning in favour of *frame-to-frame tracking* as in the object reconstruction chapter, we keep the same strategy here. Therefore the solutions following the Taylor expansion in Eq. (4.5)-(4.9) stay the same with a modification only in Eq. (4.5) and (4.6) to reflect the nested sum over voxels and volumes from Eq. (5.1).

However, if refinement is done over all already tracked frames, its convergence time will respectively increase with their number. In addition, optimization from frames separated by a large distance is not necessarily beneficial, since they may be capturing completely non-overlapping parts of the scene. Therefore, we propose to carry out *pose optimization over batches* of the last few tracked frames.

More precisely, it is done over $q \leq p$ LEVs, jointly in batches of $2b$ frames. The first half of a batch consists of frames, whose poses have already been refined once, while the second half are the lastly tracked frames. A local weighted average ϕ_{local} of these $2b$ frames is generated in each LEV. As in the object reconstruction case, each ϕ_{local} is re-calculated only on every f^{th} iteration in order to keep the objective fixed meanwhile. For stability the first $b/2$ poses are kept fixed, while each other pose is refined following the gradient descent scheme introduces in the previous chapter, resulting in one 6-element-vector update per frame. Therefore, once frame number m is tracked, optimization is carried out following the modified version of Eq. (4.12):

$$\frac{\mathrm{d}E_{geom\ LEV}}{\mathrm{d}\xi} = \sum_{\substack{volume\ \Omega_i \\ i=1..p}} \left(\sum_{voxels\ \in\ \Omega_i} \left(\phi_d\left(\xi\right) - \phi_{local}\right) \nabla_{\xi}\ \phi_d(\xi) \right) , \qquad d \in [m-2b+1, ..., m] \, . \tag{5.2}$$

Finally, the gradient descent update with step α is applied only to the second half of the batch, *i.e.* to the frames with indices $d2 \in [m-b+1, ..., m]$:

$$\xi_{d2}^{k+1} = \xi_{d2}^{k} - \alpha \frac{\mathrm{d}E_{geom\ LEV}(\xi_{d2}^{k})}{\mathrm{d}\xi} \, . \tag{5.3}$$

Concurrent GPU/CPU Processing

As our objective is a fully real-time SLAM method without any posterior processing, we execute the tracking and refinement modules concurrently. We allocate a separate GPU stream responsible for 6 DoF frame-to-frame camera pose estimation: an incoming depth map is transferred to device memory, pre-processed according to Figure 5.3, and then registered to the previous one using the limited-extent volume scheme explained above.

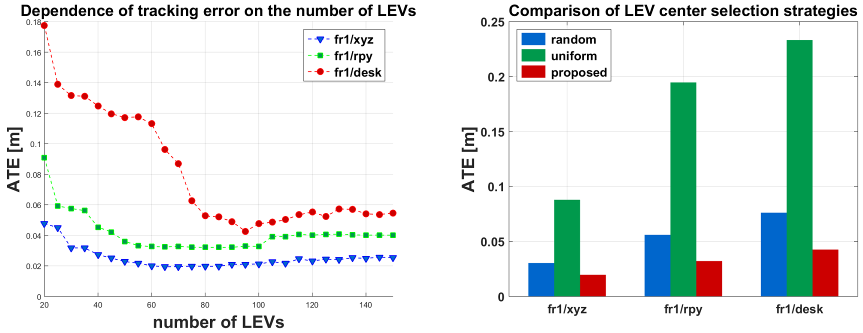
Once b frames have been processed, the CPU is signalled to start the optimization module. It carries out refinement in a *locally global* fashion: a local batch of $2b$ frames is jointly globally optimized. The batch consists of the newly tracked b poses and the b previous ones, of which the first $b/2$ are kept fixed for stability and only contribute to the weighted average calculation. When the next b frames have been tracked, the CPU is signalled to switch batches in a first in - first out fashion: the first half of the old batch is dropped, the second half is shifted to the left, and the new b frames are added at the end to fill it up completely. Then the procedure is repeated over the new batch and so forth. This strategy gives a broader context for optimization and ensures that every frame participates in the refinement twice, and is therefore geometrically consistent with frames both before and after it.

Given a trajectory estimated in this manner, a reconstruction can be generated in various ways, among which volumetric fusion [1, 158], carefully selected key-frame fusion [148], or point-based fusion [112]. As the particular method is not the focus of this work, when comparing the outputs of different pipelines we will always display results generated with the same technique, namely the publicly available fusion from PCL [1].

Implementation

Our implementation was done on the previously used Intel i7-4900MQ CPU at 2.80 GHz, and an NVIDIA Quadro K2100M GPU. Pre-processing VGA-sized depth images takes 7-8 ms: transferring the image to device memory, and estimating its normals and curvature size take approximately 4.5 ms in total, while the non-maximum suppression and sorting the peaks in order of their curvature magnitude last another 3 ms. The remaining 25 ms are entirely available for tracking, so the maximum number of iterations is set depending on the chosen number of LEVs. Depending on frame distance, 10-60 iterations are required for convergence. Refinement runs concurrently without a time limit. Instead, it switches to a new batch when it receives the signal that b new frames have been tracked.

The tracking module requires 160 MB of GPU memory for $p = 64$ SDFs (if signed distances are stored as `float` and weights as `uchar`), totalling 322.4 MB for two frames together with their depth maps. In addition, the refinement module takes 20 MB of CPU memory for the weighted average, and another 23.4 MB for 20 range images. These values demonstrate the real-time capabilities of SDF-TAR, combined with its low memory load. Furthermore, they show that there are enough resources for an additional thread, responsible for parallel data fusion.



(a) Number of LEVs.

(b) LEV anchor strategy.

Figure 5.4: **Parameter analysis of SDF-TAR**: influence of (a) the number of LEVs used and (b) the LEV anchor point selection strategy on the absolute trajectory error on sequences from the *TUM RGB-D benchmark* [208].

5.4 Evaluation

Similar to our assessment of implicit-to-implicit registration for 3D object reconstruction, we quantitatively evaluate both the trajectory estimation accuracy and the model fidelity of SDF-TAR. We first compare the performance of SDF-2-SDF and SDF-TAR on our *3D-Printed RGB-D Object Dataset*. Then we continue the evaluation of SDF-TAR on large-scale objects from the *CoRBS dataset* [233], which provides externally estimated Kinect v2 trajectories and models fused from them. Finally, to analyze our SLAM capabilities, we test on the *TUM RGB-D benchmark* [208], which contains many Axus Xtion scans of indoor spaces together with groud-truth camera trajectories.

We will again use the cloud-to-model *CloudCompare* distance to assess reconstruction accuracy. For trajectory estimation we will employ the original *RGB-D benchmark absolute trajectory error (ATE)*, which quantifies the overall error, and *relative pose error (RPE)*, which is the drift over a fixed time interval.

As before, we compare to related volumetric methods, namely KinectFusion [1] and point-to-implicit approaches [30, 35], and also to *DNA-SLAM*, which is a ToF noise-aware DVO variant [232]. We cite the error values reported in the respective papers.

Parameter Analysis

The parameters in SDF-TAR reflect the inherent properties of the environment, most of which are fixed to default values as follows. The resolution of a single LEV SDF is $8 \times 8 \times 8$ voxels, with side 8 mm for tracking and 4 mm for refinement. The grid size of 8^3 is chosen to guarantee that shared GPU memory will be efficiently utilized. Other GPU-based approaches also employ such grids [46, 161]. While the finer voxel size is advantageous for more accurate refinement, an even smaller one is not beneficial because it would become corrupted by sensor noise. The δ parameter equals the voxel size, while η is twice the voxel size, as they control the represented surface region.

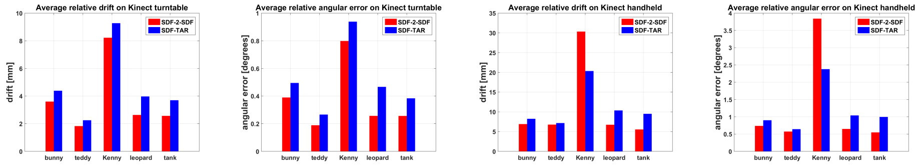


Figure 5.5: **Comparison of SDF-2-SDF and SDF-TAR** on Kinect data from the *3D-Printed RGB-D Object Dataset*. SDF-TAR decreases accuracy only slightly, while ensuring that concurrent tracking and refinement are accomplished within real-time constraints. Notably, both the translational and rotational error on the *handheld Kenny* sequence are significantly decreased.

Independent of how many LEVs are used for tracking, only $n = 8$ are used for refinement, since a good initialization is available and since generating them for a whole batch of frames on the CPU would otherwise take too much time. The batch size is 20 frames ($b = 10$), while the weighted average is generated on every $f = 5^{\text{th}}$ iteration.

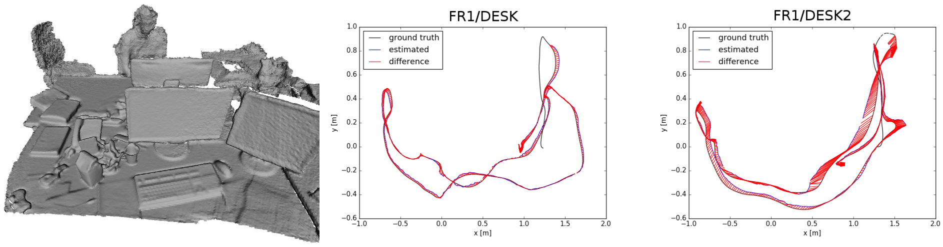
Some of the remaining parameters depend on the richness of the scanned geometry, so we investigate them here. We assess the susceptibility of trajectory estimation accuracy to changes in them on three sequences of the *TUM RGB-D benchmark* [208]: *fr1/xyz* and *fr1/rpy*, which are designed for evaluating translational and rotational motion estimation respectively, and *fr1/desk* which is a typical SLAM scenario combining both kinds of motion. In order to isolate the effect of the parameters on the partial volume registration, we disable the refinement module for this test.

First, to judge the dependence of the tracking error on the *number of LEVs*, we test with amounts from 20 to 150 per frame. The results in Figure 5.4(a) show that the error is large with only few volumes, and gradually decreases as more LEVs are taken into account. There is a rather broad range of values which lead to near-optimal results, typically around 60-90 volumes. When the LEV number becomes too high, however, the error slightly increases again. This means that the volumes have become so many that they also encompass flat regions, which inhibit registration. Naturally, in order to keep runtime as low as possible, we advocate taking the smallest amount that promises stable results, *e.g.* 80 LEVs per frame.

Next, we assess our *LEV anchor point selection procedure*, which determines where the partial SDFs are centered. We compare it to two other strategies of similar efficiency that can be applied directly over a depth map. In them the image is split into non-overlapping windows of $w \times w$ pixels, one pixel is selected per window, then back-projected to 3D and taken as the anchor point. The *uniform* approach uses the center of each window, while the *random* strategy selects a pixel at random. For all methods we first pre-process the depth map as explained in Figure 5.3 to discard invalid regions, and then test over the same number of LEVs, equal to the amount that gave optimal results for the respective sequence in the experiment above. Figure 5.4(b) shows that the uniform strategy leads to a 4-6 times higher error than our proposal, while the random sampling is nearly two times less accurate than ours. Thus our

Table 5.1: **Comparison of absolute trajectory error (ATE)** [meters] on sequences from the *TUM RGB-D benchmark* [208]. SDF-TAR achieves a considerably smaller error when the dominant motion is rotational (*e.g. fr1/rpy, fr1/360*), and demonstrates comparable accuracy under general motion.

Method	fr1/xyz	fr1/rpy	fr1/desk	fr1/desk2	fr1/360	fr1/floor
KinFu [1]	0.023	0.081	0.057	0.102	0.591	0.918
FM-pt-SDF [30]	0.021	0.042	0.035	**0.061**	0.119	0.567
FM-pt-SDF [35]	**0.014**	—	0.033	0.230	—	0.984
SDF-TAR	0.015	**0.021**	**0.030**	0.091	**0.113**	**0.279**



(a) Reconstruction of *fr1/xyz*.

(b) Estimated trajectorites.

Figure 5.6: **Examples of estimated trajectories and reconstructions by SDF-TAR** on sequences from the *TUM RGB-D benchmark* [208].

strategy clearly selects more discriminative regions that, combined with its high speed, are more advantageous for registration.

Finally, we assess the benefit of the *refinement module*. Enabling it decreases the ATE error on *fr1/xyz* by only 19%, while on *fr1/rpy* it was reduced more than 50%. Not surprisingly, on the combined motion sequence *fr1/desk* the improvement was in between: 41%. These results indicate that our refinement strategy is highly beneficial for reducing the rotational error in tracking. We attribute this to the small volumes that only encapsulate informative context around salient locations. On the contrary, motion between flat regions can only be estimated as translation-only sliding against each other, which would inhibit accurate rotation estimation.

Furthermore, we try an every-frame *refinement strategy*, whereby the same frame-to-partial model registration scheme is used, but only the last tracked pose in a batch is optimized, and the batch is switched after every frame. This refinement leads to a very slight improvement over the non-optimized trajectory. The reason is that the energy for every-frame refinement is too similar to the tracking energy, so it cannot significantly improve the pose. In contrast, the refinement approach that we follow has multiple frames influencing each other, resulting in better estimates. Thus we have developed a powerful strategy that can be applied in parallel with the tracking module and significantly reduces rotational drift.

Table 5.2: **Comparison of relative pose error (RPE) translational root-mean squared values** per frame [meters/frame] on *TUM RGB-D benchmark* [208] sequences. SDF-TAR achieves the lowest error on all examples.

Method	fr1/xyz	fr1/rpy	fr1/desk	fr1/desk2	fr1/360	fr1/floor
KinFu [1]	0.004	—	0.020	0.020	—	0.035
FM-pt-SDF [35]	**0.003**	—	0.007	0.019	—	0.050
SDF-TAR	**0.003**	**0.004**	**0.006**	**0.009**	**0.011**	**0.020**

Table 5.3: **Comparison of relative pose error (RPE) rotational root-mean squared values** per frame [°/frame] on *TUM RGB-D benchmark* [208] sequences. SDF-TAR outperforms the other methods on nearly all examples.

Method	fr1/xyz	fr1/rpy	fr1/desk	fr1/desk2	fr1/360	fr1/floor
KinFu [1]	0.474	—	2.003	1.795	—	1.718
FM-pt-SDF [35]	0.472	—	**0.759**	1.080	—	2.085
SDF-TAR	**0.442**	**1.042**	0.768	**0.993**	**1.514**	**0.844**

SDF-TAR versus SDF-2-SDF

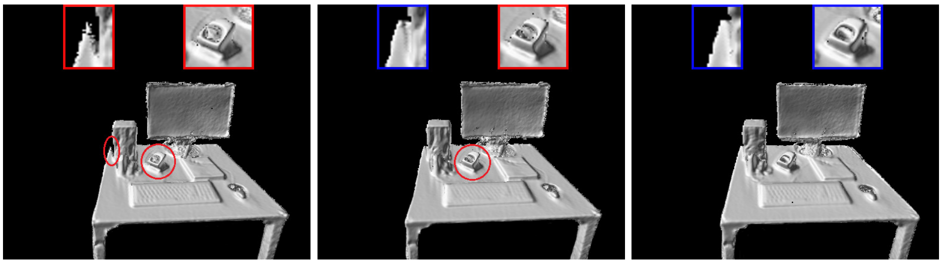
To assess the influence of the LEV modification to the standard SDF-2-SDF registration strategy, we compare our two approaches on the Kinect sequences of our *3D-Printed RGB-D Object Dataset* using the same voxel size of 2 mm. The results in Figure 5.5 confirm what could be expected: the numerical accuracy of SDF-TAR is slightly inferior to SDF-2-SDF due to the decreased density. Nevertheless, the errors remain lower than the majority of other methods examined in Figure 4.5. Moreover, the error on the challenging handheld *Kenny* sequence is significantly decreased when taking only LEVs rather than the entire projective SDF. Therefore, SDF-TAR is a promising modification of SDF-2-SDF that is capable of applying our dense implicit-to-implicit energy to large spaces and SLAM scenarios in real time, executing both tracking and refinement concurrently.

Simultaneous Localization and Mapping

We continue our quantitative evaluation on one of the most widely used publicly available datasets, the *TUM RGB-D benchmark* [208], and therefore now assess the SLAM capabilities of SDF-TAR. The absolute and relative tracking errors are summarized in Tables 5.1, 5.2 and 5.3, while Figure 5.6 shows examples of estimated trajectories and reconstructions. The ATE testifies that SDF-TAR considerably outperforms related works on sequences with dominant rotational motion, and achieves on-par or better accuracy on general types of motion. Moreover, our relative rotational drift is well below 1° even on the challenging *fr1/floor* sequence. We, therefore, conclude that the LEVs reduce the negative influences of noise, blur and rolling shutter effect by constraining registration to the most discriminative local geometry, and effectively avoiding regions that typically impede accuracy, such as flat surfaces.

Table 5.4: **Comparison of relative pose error (RPE) root-mean squared values** for translational [meters/second] and rotational [°/second] error per second on sequences from the *CoRBS dataset* [233].

Method	Desk D1		Cabinet E1		Human H1	
	transl.	rot.	transl.	rot.	transl.	rot.
DNA-SLAM [232]	0.027	0.970	0.035	1.426	**0.020**	**0.725**
KinFu [1]	**0.026**	1.739	0.045	1.047	0.034	1.626
FM-pt-SDF [33]	0.032	1.753	0.033	1.731	0.041	1.891
SDF-TAR	0.030	**0.964**	**0.032**	**0.990**	0.037	1.456



(a) KinFu [1]. (b) FM-pt-SDF [33]. (c) SDF-TAR.

Figure 5.7: **Qualitative comparison on *Desk1* from the *CoRBS dataset*** [233]: related approaches wash out fine structures due to drift (marked in red), while the concurrent refinement of SDF-TAR reduces it, yielding more detailed, higher fidelity results.

Reconstruction of Large Objects

Finally, we assess the performance of SDF-TAR on the task of reconstructing large-scale objects, such as furniture items and industrial machines. The difference in scanning motion between SLAM and object reconstruction of any scale is that the latter is usually executed with an outside-in motion facing the object, while the former is done in an inside-out manner which is typically more prone to tracking errors [36]. Thus it is interesting to see if our approach generalizes well to both kinds of motion.

As mentioned, we make use of the *CoRBS dataset* [233], but since it is relatively new and was created after KinectFusion [157] and the publication Canelhas *et al.* [35], we run KinFu [1] and the ROS version of FM-pt-SDF by Canelhas [33] ourselves. For all tests we used a voxel size of 8 mm, while other parameters were set to the most advantageous ones defined by the respective authors of each approach. In addition, we include results from DNA-SLAM [232], which is a SLAM system from the authors of the *CoRBS dataset*, specifically designed for time-of-flight cameras, but, unfortunately, reporting only RPE values and no model errors.

Table 5.4 provides an overview of the relative trajectory errors per second. KinFu [1] and FM-pt-SDF [33] perform similarly, as was often the case for smaller-scale objects, while DNA-SLAM and our SDF-TAR achieve higher precision. In some cases DNA-SLAM outperforms us, since it is specifically de-

Table 5.5: ***CloudCompare* absolute cloud-to-model error comparison** [centimeters] on objects from the *CoRBS dataset* [233].

Method	Desk D1	Cabinet E1	Human H1
KinFu [1]	1.5686	1.2504	0.7105
FM-pt-SDF [33]	1.3266	1.1599	**0.6583**
SDF-TAR	**0.9856**	**1.0552**	0.7258

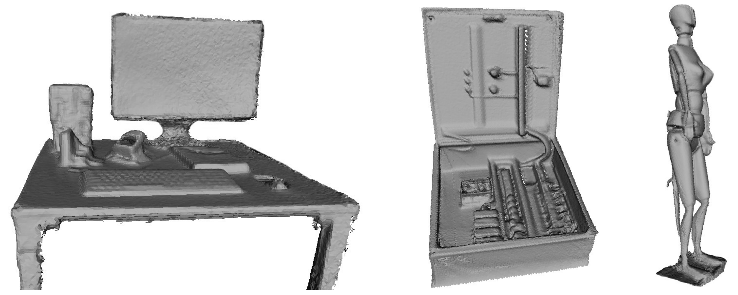


Figure 5.8: **SDF-TAR reconstructions of large objects from the *CoRBS dataset*** [233].

signed for this kind of depth sensor. Nevertheless, SDF-TAR still demonstrates excellent rotational motion estimation.

The *CloudCompare* results in Table 5.5 exhibit a similar trend. We achieve the smallest model error on most objects, which we attribute to the smaller rotational drift, combined with the benefit of online refinement. This proves that SDF-TAR has successfully adapted SDF-2-SDF registration to larger volumes of interest, as can be seen from the reconstructed models shown in Figure 5.8.

We provide a further qualitative comparison in Figure 5.7. Even though all methods manage to recover a consistent model, the zoomed-in parts show that larger drift causes the other approaches to wash out finer structures, such as the sides of the stapler, while we manage to keep them intact.

These results confirm that the SDF-2-SDF energy is well suited both for small- and large-scale object reconstruction, in which the motion is object-centered. It is also applicable to more challenging outward-facing SLAM scanning, where it is especially accurate under rotational motion.

5.5 Conclusion

We have presented a hybrid GPU/CPU system for concurrent tracking and batch refinement that extends out SDF-2-SDF registration scheme to large volumes of interest. For this purpose we developed a novel memory reduction scheme, which aligns multiple voxel grids representing partial SDFs anchored at locations of distinctive geometry. These limited-extent volumes not only provide an easy to implement way for keeping memory load and runtime fixed, but also lead to considerably more accurate rotational motion estimation than related methods, as demonstrated on public datasets.

Even though we are now able to handle volumes of almost arbitrary dimensions, this still does not mean that we can accurately capture our surrounding 3D world. The main reason for this is that so far we have assumed that this world is static. While this is true when we scan household objects made out of hard plastic or empty office spaces, our surroundings are actually dynamic. Things change shape over time like a gradually deflating balloon or a growing plant. Even faster than these processes are interactions that happen every second: people interact with each other and with the objects around them. These problems are much more challenging than 6 DoF camera pose estimation, because essentially any point may have moved to any other location. As the goal that we set out at the beginning of this thesis was to capture the real world, we now investigate how to adapt our implicit-to-implicit strategy to more general non-rigid motion.

Part IV

Deformable 3D Reconstruction

6 Variational SDF Evolution

We now aim to reconstruct non-rigidly moving 3D objects with our implicit-to-implicit energy. In particular, the scenario of interest involves using a single RGB-D camera, which is either static or moving while capturing non-rigid motion in real time. Moreover, we want to reconstruct arbitrary scenes in which there may be more than one subject, such as two people interacting or a person handling an object. Therefore we have to be able to manage the resulting topological changes that occur when surfaces merge and split, *e.g.* when people shake hands or a person takes off their jacket. In addition, the movements might be slow or fast, causing small or large pose differences, and exposing different amounts of new geometry in each frame. Finally, we want our method to be general and therefore independent of shape priors or templates. To sum up, our non-rigid 3D reconstruction approach has to manage:

- fast, unconstrained motion;
- multiple interacting subjects;
- topological changes;
- without prior knowledge.

In the following we discuss how to achieve these objectives with appropriate modifications to our implicit-to-implicit scheme. As opposed to recent techniques that achieve impressive results by considering the problem from a SLAM perspective and putting emphasis on a deformation field that brings depth data into alignment, we view the task as *shape evolution*. A suitable *analogy* here is to imagine the shape as a piece of clay. We may play with it and deform it into any other shape that can be constructed with the same amount of clay, without requiring that the points that were initially on the surface of the blob of clay are still on the surface in the end structure. We may even deform a stick into a donut, or into a pair of smaller sticks. In our reconstruction scenario we know that while two consecutive frames might be separated by a large pose change, they are not as dramatically different as the examples above. This provides us with constraints to appropriately stop the shape evolution so that consecutive frames can be fused together into a geometrically consistent reconstruction after the non-rigid motion is factored out.

The framework that we propose is based on the variational level set method [256]. Knowing that SDFs inherently handle topological changes, which is one of our main goals, we aim to warp a given initial SDF to a target SDF via gradient flow without explicit correspondence search. We keep the data term of our energy to the implicit-to-implicit alignment one used throughout this thesis. Here it is not subject to a 6 DoF transformation, but to a much higher-dimensional warp field, so regularization terms are required. To ensure geometrically consistent reconstructions, we devise and compare different strategies. In particular, we use an approximately Killing vector field regularizer that is similar to an as-rigid-as-possible [201]. Alternatively, we apply gradient flow in Sobolev space [154], which is smoother and permits coarse-to-fine evolution that first recovers global deformations and then adds smaller-scale changes. In this way we are eventually able to capture rapid motions, topological changes and interacting agents.

As always, we verify the performance of our approaches through qualitative and quantitative assessment. A major issue in single-stream non-rigid reconstruction is the lack of real-world datasets that permit quantitative evaluation. We address this problem by using mechanical toys that have a rest pose, in which they can be accurately reconstructed to obtain a groud-truth 3D model. After non-rigid-movement sequences are recorded and reconstructed, we can compare our output to the initially generated model. While this does not permit every-frame evaluation, it is a first effort towards quantifying the performance of non-rigid reconstruction approaches, so we make our data publicly available.

6.1 Introduction

As we have seen in the last two chapters, the wide availability of off-the-shelf RGB-D sensors and the growing popularity of virtual and augmented reality have made 6 DoF camera pose estimation and volumetric fusion for real-time single-stream 3D reconstruction possible. Many techniques for capturing static environments have demonstrated impressive results [39, 110, 114, 157, 161, 239, 258]. However, real-life scenes also include moving people, interacting with objects in their surroundings and with each other. This requires the capture of non-rigidly moving surfaces, which is a very unconstrained problem that still poses major challenges.

The problem is ill-posed because there are infinitely many solutions that may have deformed one frame to the next [74]. While older techniques resorted to the use of multiple cameras [31, 44, 55, 54, 107] or templates [19, 95, 131, 261] in order to better constrain the solution space, nowadays methods that utilize a single RGB-D camera are emerging. DynamicFusion [156] first demonstrated real-time simultaneous tracking and reconstruction of non-rigid surfaces. Several works build over it, incorporating colour features [98], albedo constraints [81] or human-specific priors [247, 248]. Their results are of ever-improving visual quality, however, they are still constrained mainly to contrived

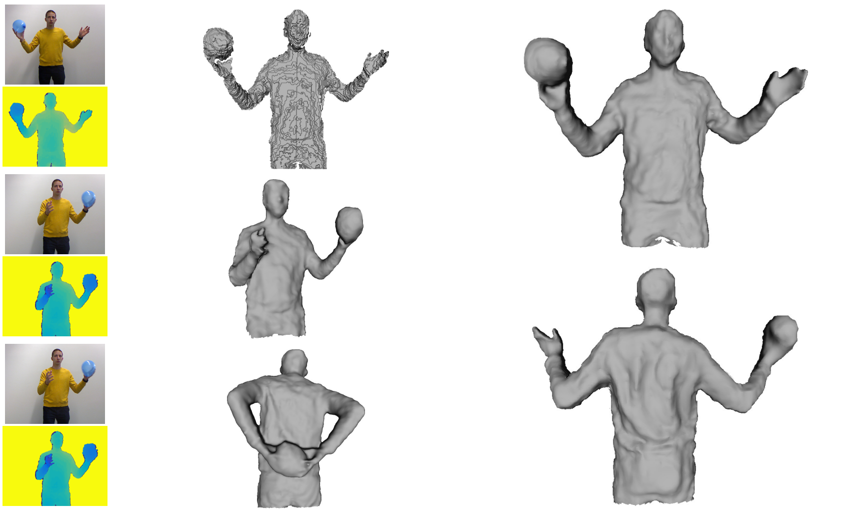

(a) Input. (b) Warped live frames. (c) Canonical-pose reconstruction.

Figure 6.1: **Non-rigid reconstruction of a person playing with a balloon**. (a) Our system takes a single depth stream as input and warps each frame towards the canonical model in order to grow it. (b) Then the model is warped back towards the live depth for display to the user. (c) The final output is a complete 3D model despite the topological changes that occurred.

motion without interactions or topological changes.

Having studied the power of implicit-to-implicit registration in the rigid case, and knowing the advantageous properties of SDFs under changing topology, we addresses these issues through the use of SDF evolution. This is the process of gradually deforming one SDF to another one under variational gradient flow. The majority of recent approaches for both dynamic and static reconstruction employ a SDF for storing the growing reconstruction [98, 157, 156]. One of the main advantages of this representation is its ability to smooth out noise when repeated measurements at the same voxel are averaged [45]. However, these methods intermittently revert back to a mesh representation in order to estimate correspondences for non-rigid alignment, thereby losing accuracy, computational speed and the SDF capability to conveniently capture topological changes.

Therefore we propose a method that operates entirely within the SDF representation. It warps an initial SDF to a target SDF via gradient flow without correspondence search, steered by a data term that imposes voxel-wise alignment. Furthermore, we propose two strategies that ensure geometric plausibility. On the one hand, we include an approximately Killing vector field [200] energy term which enforces the estimated deformation field to generate locally nearly isometric motions, acting similar to an as-rigid-as-possible regularizer [201]. On the other hand, instead of adhering to the commonly used gradient defined via an L^2 inner product, we apply gradient flow defined in Sobolev space [154], which acts as a pre-conditioner ensuring a coarse-to-fine evolution behaviour [211]. While the former approach [193] is slightly faster and thus allows for the incorporation of additional terms which impose desirable geometric properties, such as unit gradient magnitude, the latter one [195] achieves higher geometric detail without over-smoothing effects. As a result, our variational solution is able to handle challenging scenarios such as changing topology and fast motion. Figure 6.1 shows an example of our novel approach that:

- stays entirely within the SDF representation, circumventing intermittent conversion to a mesh;
- gradually evolves a shape without explicitly estimating correspondences;
- handles topological changes and large, rapid movements.

6.2 Related Work

Here we discuss existing approaches on level set evolution, vector field estimation and deformable surface tracking in RGB-D data, identifying their limitations in the context of our problem of interest and suggesting remedies.

Multi-view and template-based surface tracking External constraints help to alleviate the highly unconstrained nature of non-rigid registration. For example, the system of Zollhöfer *et al.* [261] deforms a template to incoming depth frames in real time, but requires the subject to stay absolutely still during the template generation, which cannot be guaranteed when scanning animals or kids. Multi-camera setups are another way to avoid the challenging task of incrementally building a model. For instance, Fusion4D [55] recently demonstrated a powerful real-time performance capture system using 24 cameras and multiple GPUs, which is a setup not available to the general user. Moreover, Section 8 of the publication states that even though Fusion4D deals with certain topology changes, the algorithm does not address the problem intrinsically. We explicitly tackle this issue here, but as the focus is on reconstructing a dynamic environment using a single RGB-D sensor without any prior knowledge, we will not discuss other systems that employ specialized multi-camera set-ups [4, 44, 107], hand [214, 216], face [218], skeleton [247] or human body [19, 248] priors, or that require the acquisition of a static template. We refer the reader to the recent comprehensive overview by Zollhöfer *et al.* [263] for an extensive analysis of the properties of such methods.

Single-stream incremental non-rigid reconstruction Template-free methods for non-rigid fusion from a single depth camera have been on the rise since 2015 with the development of the offline bundle adjustment scheme of Dou *et al.* [56] and the first real-time solution for simultaneous surface tracking and reconstruction, DynamicFusion [156]. Several extensions to this seminal work have been proposed, most notably VolumeDeform [98] which combines the used dense depth-based correspondences with sparse SIFT features to reduce drift and handle tangential motions, and the system of Guo *et al.* [81] which increases robustness by integrating surface albedo constraints. Nevertheless, they have been demonstrated only on examples of relatively constrained motions without changing topology.

Level set methods Fast motion, surface merging and splitting are inherently handled by the signed distance field representation [163]. It has been applied for segmentation [9, 87] and registration [125, 144] in medical imaging, where organ shape priors are typically available, and for surface manipulation and animation on complete noise-free models in graphics [43, 73, 223, 236]. In computer vision Paragios *et al.* [165] and Fujiwara *et al.* [74] have used level sets for non-rigid registration on 2D image data and have discussed extensions to 3D. The task of fusion from 2.5D data is more challenging since new data has to be incremented in a consistent manner.

Scene flow and piecewise rigid motion The step before fusion requires estimating a dense warp field between a new frame and the existing reconstruction. This is the objective in scene flow [97, 171, 227, 231, 234]. Related to these are also approaches that segment the scene into static and dynamic components and reconstruct them separately [103, 179]. Many of these techniques are variational in nature, combining a data term that imposes similarity between the warped observed data and the target model, and a regularizer that imposes motion smoothness to better constrain the solution space. We thus propose to extend the variational level set method [256] to the setting of incremental fusion from a single depth stream.

As the challenge is how to increment new observations instead of erroneously registering them to old data, we investigate additional regularizers. Non-rigid motion tends to be not only smooth, but also volume-preserving. Therefore a prior that enforces the field to be solenoidal, *i.e.* divergence-free, would benefit the fusion. Killing vector fields are of this class and generate locally isometric motions [12, 200, 215]. Thus they offer a way to impose a rigidity prior directly through the warp field, rather than resorting to embedded deformation [209] or as-rigid-as-possible schemes [201].

Gradient flow Another important remark is that the L^2-type inner product employed for gradient flow in most variants of the variational level set method [163, 164, 256] assumes a metric that may lead to slow convergence and sub-optimal solutions [212]. Instead, gradient flow in the Sobolev space H^1 has been shown to have a superior performance without changing the global optimum thanks to a desirable coarse-to-fine evolution behaviour that is robust to spurious artifacts [212]. We refer the reader to the book of Neuberger [154] for a thorough mathematical introduction to the topic.

6.3 SDF Evolution

In the non-rigid case the unknown Υ from Eq. (3.11) is a deformation field that brings the two SDFs in voxel-wise alignment. Thus our objective is to determine a vector flow field $\Psi = (U, V, W) \colon \mathbb{N}^3 \to \mathbb{R}^3$ of the same resolution as the SDFs. U, V and W denote its x-, y- and z-components respectively, each of which is a scalar grid $\mathbb{N}^3 \rightarrow \mathbb{R}$. We denote the vector applied at voxel (x, y, z) by (u, v, w).

In the non-rigid case the 3D reconstruction is typically accumulated in the first pose that was observed, which is called the canonical pose. The task is to factor out the motion in every frame so that it can be fused in a geometrically consistent manner with the canonical model. Therefore a frame-to-model alignment strategy is most appropriate for our implicit-to-implicit scheme here.

Given the current state of the cumulative model ϕ^{i-1}_{model} and an incoming RGB-D pair (I^i_{RGB}, I^i_D), we iteratively estimate a deformation field that warps the projective TSDF ϕ^i_{proj} generated from I^i_D towards ϕ^{i-1}_{model}, resulting in the warped TSDF ϕ^i_{warped}. Then we fuse ϕ^i_{warped} into the global model, obtaining its updated state ϕ^i_{model}. Finally, we run a backward deformation from ϕ^i_{model} towards ϕ^i_{proj} in order to provide a live visualization to the user.

We assume that both the scene and the camera are moving. Therefore we estimate a rigid camera transformation using our SDF-2-SDF scheme from Chapter 4. We prefer this formulation over ICP variants [15, 180], since they would need a very robust norm to discard the many outliers that result from large deformations.

Next, we describe our variational model for non-rigid 3D reconstruction from a single depth stream.

Signed Distance Field Evolution Energy

As a new RGB-D frame is acquired and we estimate the approximate camera pose, we generate its projective TSDF ϕ_{proj}. Next, we iteratively warp it towards the canonical TSDF ϕ_{model}. In iteration t, we calculate a deformation field increment $\Psi = (U, V, W)$ and apply it to the current warped TSDF $\phi^{(t)}_{proj}$, obtaining its new state $\phi^{(t+1)}_{proj}$ via tri-linear interpolation. We do this following a variational formulation consisting of a data term and a combination of regularizers:

$$E_{def}(\Psi) = E_{data}(\Psi) + w_{reg} E_{reg}(\Psi) \,, \tag{6.1}$$

where $w_{reg} > 0$ controls the trade-off between data fidelity and regularity. A solution of this model can be found via a gradient descent scheme with step size $\alpha > 0$:

$$\Psi^{(t+1)} = \Psi^{(t)} - \alpha \; \nabla E_{def}\left(\Psi^{(t)}\right) \,, \tag{6.2}$$

where $\nabla E_{def}\left(\Psi^{(t)}\right)$ denotes the variational derivative of the energy with respect to the deformation field. As will be explained in Section 6.3.2, ∇E_{def} depends on the choice of the underlying inner product, but beforehand we define our deformation energy terms.

Data term

Our data term is driven by the intuition that under perfect alignment, the warped and the target TSDFs will have identical signed distance values in each overlapping voxel. Therefore the value at each voxel (x, y, z) of the current frame ϕ_{proj}, displaced by its flow vector (u, v, w), will be equal to the value in that voxel in ϕ_{model}. Thus to obtain the warp, we minimize the direct squared voxel-wise difference:

$$E_{data}(\Psi) = \frac{1}{2} \sum_{x,y,z} \left(\phi_{proj}(x+u, y+v, z+w) - \phi_{model}(x, y, z)\right)^2 . \tag{6.3}$$

We obtain the derivative by standard calculus of variations:

$$\nabla E_{data}(\Psi) = \left(\phi_{proj}(\Psi) - \phi_{model}\right) \nabla \phi_{proj}(\Psi) . \tag{6.4}$$

Note that we use the symbol ∇ both for the spatial gradient of ϕ and for the variational derivatives of the energy terms. The derivations of this and all following formulas are given in Appendix A.

Regularization

Commonly, non-rigid registration methods impose regularity constraints in order to introduce additional information, thereby reducing the solution space of the problem [263]. In our setting regularity can be enforced through the warp field itself, as well as over the TSDFs. We propose several alternatives in this section and analyze how to best combine them for efficient deformable reconstruction.

6.3.1 Damped Approximately Killing Vector Field Regularizer

Uniform motion The expected input to our system is noisy Kinect data, which might cause inconsistencies within voxel neighbourhoods that result in holes in the reconstruction. A classical Tikhonov-type regularizer can be used to reduce spurious artifacts and impose motion smoothness, as often done in scene and optical flow [28, 97, 234]:

$$E_{smooth}(\Psi) = \frac{1}{2} \sum_{x,y,z} \left(|\nabla U(x,y,z)|^2 + |\nabla V(x,y,z)|^2 + |\nabla W(x,y,z)|^2\right) . \tag{6.5}$$

Using calculus of variations we obtain:

$$\nabla E_{smooth}(\Psi) = -(\Delta U, \Delta V, \Delta W)^\top , \tag{6.6}$$

where ΔU denotes the Laplace operator applied to the x-component of the flow field, and similarly for V and W.

Divergence-free flow Another strategy is to prevent uncontrollable deformations via rigidity constraints. Most common are the as-rigid-as-possible [201] and embedded deformation [209] formulations, which ensure that the vertices of a latent control graph move in an approximately rigid manner. Here we propose an alternative, whereby local rigidity is imposed directly through the deformation field.

A 3D flow field that generates locally isometric motions is called a *Killing vector field* [12, 200, 215], named after the German mathematician Wilhelm Killing. It is divergence-free, *i.e.* volume-preserving, and satisfies the *Killing condition* $J_\Psi + J_\Psi^\top = \mathbf{0}$, where J_Ψ is the Jacobian of the field. However, it does not regularize angular motion.

A field which generates only nearly isometric motion and thus balances both volume and angular distortion is an *approximately Killing vector field (AKVF)* [200]. It minimizes the Frobenius norm of the Killing condition:

$$E_{akvf}(\Psi) = \frac{1}{2} \sum_{x,y,z} \left\| J_\Psi + J_\Psi^\top \right\|_F^2 . \tag{6.7}$$

Its functional derivative is:

$$\nabla E_{akvf}(\Psi) = -2(\Delta U, \Delta V, \Delta W)^\top - 2\left(\frac{\partial(\mathrm{div}\Psi)}{\partial x}, \frac{\partial(\mathrm{div}\Psi)}{\partial y}, \frac{\partial(\mathrm{div}\Psi)}{\partial z} \right)^\top , \tag{6.8}$$

where $\mathrm{div}\Psi = U_x + V_y + W_z$ is the divergence of the warp field. We refer the reader to the supplementary material for complete derivations of all equations in this section.

However, this constraint might be too strict for surfaces undergoing large deformations. Thus we propose to damp the Killing condition. First, we rewrite Eq. (6.7) using the column-wise stacking operator $vec(\cdot)$ as follows:

$$\begin{aligned} E_{akvf}(\Psi) &= \frac{1}{2} \sum_{x,y,z} vec(J_\Psi + J_\Psi^\top)^\top vec(J_\Psi + J_\Psi^\top) = \\ &= \sum_{x,y,z} vec(J_\Psi)^\top vec(J_\Psi) + vec(J_\Psi^\top)^\top vec(J_\Psi) . \end{aligned} \tag{6.9}$$

Next, we notice that the first term can be written as:

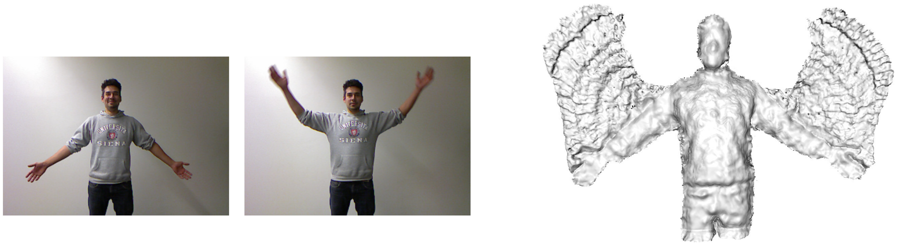
$$vec(J_\Psi)^\top vec(J_\Psi) = |\nabla U|^2 + |\nabla V|^2 + |\nabla W|^2 = 2E_{smooth}(\Psi) . \tag{6.10}$$

Therefore we devise our *damped Killing regularizer* as a damped-down AKVF condition, in which more weight is given to the motion smoothness component:

$$E_{Killing}(\Psi) = \sum_{x,y,z} \left(vec(J_\Psi)^\top vec(J_\Psi) + \gamma vec(J_\Psi^\top)^\top vec(J_\Psi) \right) . \tag{6.11}$$

The parameter γ controls the trade-off between Killing property and motion uniformity. A value of $\gamma = 1$ corresponds to the AKVF condition from Eq. 6.7. The respective derivative is:

$$\nabla E_{Killing}(\Psi) = -2(\Delta U, \Delta V, \Delta W)^\top - 2\gamma \left(\frac{\partial(\mathrm{div}\Psi)}{\partial x}, \frac{\partial(\mathrm{div}\Psi)}{\partial y}, \frac{\partial(\mathrm{div}\Psi)}{\partial z} \right)^\top . \tag{6.12}$$



Motion. Reconstruction.

Figure 6.2: **Effect of faulty gradient flow**. If the magnitude of the level set is not conserved to be one, the reconstruction accumulates artifacts.

Level set property One of the characteristic properties of a signed distance field is that its gradient magnitude equals unity everywhere where it is differentiable [163]. To ensure geometric correctness during the evolution of ϕ_{proj} towards ϕ_{model}, this property has to be conserved [129]:

$$E_{\underset{set}{level}}(\Psi) = \frac{1}{2} \sum_{x,y,z} \left(|\nabla \phi_{proj}(x+u, y+v, z+w)| - 1 \right)^2 . \tag{6.13}$$

Again, applying the calculus of variations we obtain:

$$\nabla E_{\underset{set}{level}}(\Psi) = \frac{|\nabla \phi_{proj}(\Psi)| - 1}{|\nabla \phi_{proj}(\Psi)|_\epsilon} \, H_{\phi_{proj}(\Psi)} \, \nabla \phi_{proj}(\Psi) \, , \tag{6.14}$$

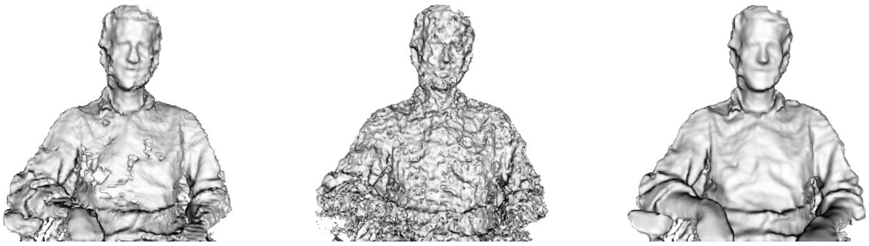
where $H_{\phi_{proj}(\Psi)} \in \mathbb{R}^{3\times3}$ is the currrent TSDF's Hessian matrix, composed of second-order partial derivatives. To avoid division by zero we use the expression $|\cdot|_\epsilon$, which equals the norm plus a small constant $\epsilon = 10^{-5}$.

Without preserving the level set property, a faulty gradient is propagated. As shown in Figure 6.2, this leads to artifacts, which may sometimes look like *Victoria's secret*$^{\text{TM}}$ wings, but are not always desirable.

This term is not only suitable for imposing regularity over the warped TSDF, but also for reducing noise in it, since spurious artifacts will get smoothed out when this constraint is applied. However, it does not hold strictly on a discretized signed distance field with a numerically approximated gradient [163], and is not valid at the border of voxel truncation, so it may lead to over-smoothing effects. To overcome these issues, we instead consider pre-conditioning the gradient flow, as explained next.

6.3.2 Sobolev Gradient Flow

The concept of Sobolev gradient flow was developed several decades ago in the context of the numerical solutions of partial differential equations. The main idea is to compute the variational derivative of an energy with respect to the inner product of a smooth subspace of L^2, *i.e.* a Sobolev space, in order to obtain a gradient, which employed in a descent scheme yields a gradient flow that favours globally consistent solutions and is less susceptible to undesired



(a) $s = 3$ (b) $w_{smooth} = 0$ (c) default

Figure 6.3: **Parameter analysis for** $E_{defSobolev}$: (a) a small neigbourhood s is not able to fully overcome the effects of noise; (b) no motion regularization results in inconsistent geometry; (c) the default setting $s = 7$, $w_{smooth} = 0.2$, $\lambda = 0.1$ yields a pleasing reconstruction.

local minima. To describe this effect Sundaramoorthi *et al.* [211] coined the term *coarse-to-fine evolution*, which accurately summarizes the fact that coarse-scale changes are favoured over fine-scale ones. In the context of incremental 3D reconstruction, this means that the warped TSDF will first adapt to more global deformations before eventually converging also with respect to fine-scale details.

To compute a Sobolev gradient, it is sufficient to project the original gradient ∇E_{def} to the Sobolev space H^1 [32]. As done in traditional descent schemes, let us define ∇E_{def} from Eq. (6.2) as the L^2 gradient $\nabla_{L^2} E_{def}$. Thus we obtain:

$$\nabla_{H^1} E_{def} = (Id - \lambda\Delta)^{-1} \, \nabla_{L^2} E_{def} \, , \tag{6.15}$$

where Id denotes the identity operator. Eq. (6.15) involves the solution of an equation system, but it is possible to derive an approximate way of obtaining Sobolev gradients. First we note that Eq. (6.15) can be realized via

$$\nabla_{H^1} E_{def} = S * \nabla_{L^2} E_{def} \, , \tag{6.16}$$

where the filter S is the impulse response of the operator $(Id - \lambda\Delta)^{-1}$. In practice, we approximate S for chosen λ and filter size s by solving the following system:

$$(Id - \lambda\Delta) S = v \, , \tag{6.17}$$

where v is a one-hot vector that corresponds to a discretized Dirac impulse of size $s \times s \times s$ voxels, and Δ is the Laplacian matrix discretized via a s-point finite-difference stencil.

However, 3D convolutions might become prohibitively expensive for large values of s. Thus we further approximate the Sobolev kernel S by three separable 1D convolutions. To do so, we calculate the tensor higher-order SVD decomposition [119] of S and retain only the first singular vector from each resulting U matrix, and after normalization to unit sum obtain the 1D s-element filters S_x, S_y and S_z. Note that as their entries are identical, the subscript is used to denote the spatial direction of application. This is an approximation of S with crucial performance advantages. The process of generating the separated kernels is outlined in more details in Appendix A.

Combined Energy

While any of the energy terms discussed in Section 6.3 can be combined into E_{reg} with appropriate balancing weights, and the proposed Sobolev filters can be additionally applied to regularize any energy, each of these components entails an increase in runtime. As we aim for applications at interactive rates, we favour two of the possible combinations.

If we are to use Sobolev gradient flow, a regularizer that imposes smooth motion is sufficient, since the gradient descent will follow a coarse-to-fine evolution that will first recover global motion and then add details. The following energy drives the gradient flow in our non-rigid 3D reconstruction method called *SobolevFusion*:

$$\nabla E_{defSobolev} = \nabla_{H^1}(E_{data} + w_{smooth} E_{smooth}) \,. \tag{6.18}$$

As the Sobolev gradient flow enforces globally consistent motion without changing the global optimum [212], we do not need to impose additional rigidity constraints or carry out level set re-initialization [129, 130].

However, if the kernel size s is too large, the execution time starts to lag behind near-real-time rates. Therefore we propose another alternative, without Sobolev regularization, which allows for incorporation of more priors into the energy formulation. Due to the lack of pre-conditioning, we need to impose rigidity constraints and ensure that the level set property is conserved throughout the evolution. The energy below is used for the version of our variational SDF evolution approach called *KillingFusion*:

$$\nabla E_{defKilling} = \nabla_{L^2}(E_{data} + w_k E_{Killing} + w_{ls} E_{\substack{level\\set}}) \,. \tag{6.19}$$

As our experiments will demonstrate, the two strategies lead to similar results. While $E_{defKilling}$ is slightly faster, $E_{defSobolev}$ does not suffer from over-smoothing effects and may yield reconstructions with better geometric details.

Parameter Analysis

We use the *Andrew-Chair* full-loop sequence from Dou *et al.* [56] in order to determine the most advantageous parameters in case of using Sobolev pre-conditioning with $E_{defSobolev}$, shown in Fig. 6.3. Our model is robust with regard to the parameter choice and achieves good results with a variety of settings, of which we recommend neighbourhood size $s = 7$, filter parameter $\lambda = 0.1$ and motion smoothness $w_{smooth} = 0.2$ as default.

A Sobolev filter size $s = 3$ is not sufficient to achieve satisfactory results. While a larger kernel would impede the speed, the differences with $s \geq 7$ become negligible.

The parameter λ has an effect on the convergence rate. We estimated empirically that doubling its value reduces the number of iterations by 3-8%. Moreover, motion regularity is essential to overcome noise. The ranges $\lambda \in [0.05; 0.4]$ and $w_{smooth} \in [0.1; 0.5]$ yield high fidelity reconstructions, so we set the default values as the midpoints of those intervals.

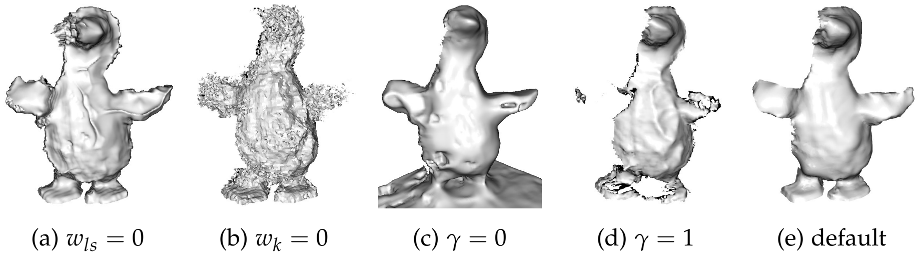

(a) $w_{ls} = 0$ (b) $w_k = 0$ (c) $\gamma = 0$ (d) $\gamma = 1$ (e) default

Figure 6.4: **Parameter analysis for** $E_{defKilling}$: (a) no level set property preservation; (b) no motion regularization; (c) conventional motion smoothness without a Killing component; (d) pure AKVF condition; (e) default setting: $w_{ls} = 0.2$, $w_k = 0.5$, $\gamma = 0.1$.

For the case without Sobolev regularization, we use the fast-motion *Duck* sequence from the *Deformable 3D Reconstruction Dataset* of KillingFusion, since the effect of the damped Killing regularizer is better observable under large motion. As shown in Fig. 6.4 without level set property preservation the model is not smooth and develops fine-scale artifacts where the property has been violated during the evolution. If all motion regularizers are disabled, the moving parts of the object, such as its wings and head, get destroyed as more frames are fused inconsistently. If only E_{smooth} is used as motion regularization, the reconstruction is somewhat smoother, but holes appear in several regions due to discrepancies. Conversely, if no damping is applied to the AKVF condition, the stronger rigidity prior causes the non-rigidly moving wings to nearly vanish. Our default setting of $w_{ls} = 0.2$, $w_k = 0.5$, $\gamma = 0.1$ yields a geometrically consistent reconstruction. We empirically determined the suitable range for γ to be $[0.05; 0.25]$.

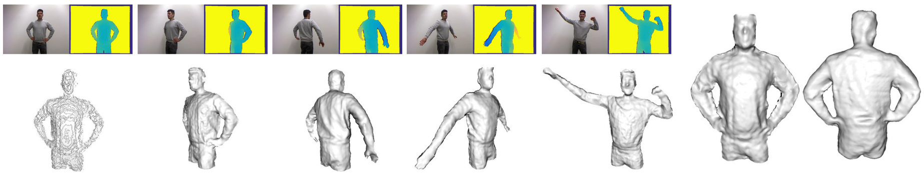
In all tests we used a gradient descent step size $\alpha = 0.1$.

Implementation

One of the main benefits of our correspondence-free variational energy formulation is that it can be applied to each voxel independently, so all displacement vector updated can be computed in parallel. We tested our implementations on a laptop with an Nvidia Quadro K1100M GPU with 2 GB of global memory, and on a desktop PC with an Nvidia Titan Black with 6 GB of memory. Depending on the bounding volume, we used a voxel size in the range 4-12 mm in order to fit the entire regular voxel grid into GPU memory.

On the laptop we achieve 30 frames per second for 64^3 voxels with $E_{defSobolev}$ and for 80^3 voxels with $E_{defKilling}$. On the PC the resolution is approximately doubled, with real-time performance for 128^3 and 150^3 voxels respectively. The runtime with Sobolev regularization can be improved if a smaller kernel size is used, at the risk of certain loss of geometric quality. In particular, a neighbourhood of $s = 5$ achieves similar speed to the L^2-energy formulation.

Details on the fast implementation of separable Sobolev kernels are given in Appendix A, alongside with the derivations of all formulas from this section.



(a) Warped live frames. (b) Canonical model.

Figure 6.5: **Non-rigid reconstruction from a single depth stream using the damped AKVF regularizer**. We obtain a geometrically consistent model after a 360° loop under topological changes and large motion.

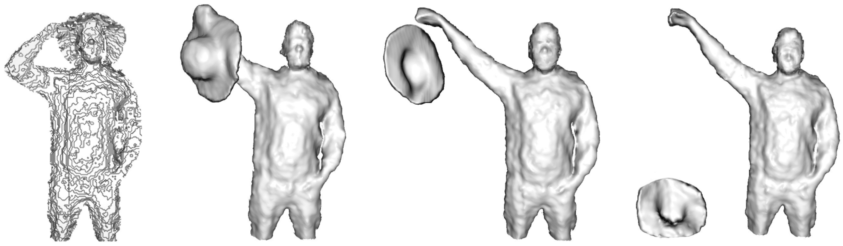


Figure 6.6: **Warped live frames from a sequence of a person taking off their hat**. The resulting topological changes are handled seamlessly.

6.4 Evaluation

In this section we carry out various tests of the non-rigid reconstruction and voxel correspondence components of the proposed formulation. We qualitatively and quantitatively compare to state-of-the-art methods. As mentioned, the outputs of the two versions of our variational approach, *i.e*. with Sobolev gradient flow, *SobolevFusion*, and with damped AKVF constraints, *KillingFusion*, are similar. Thus we will show several examples of each, but include the results of both systems only if they are notably different, and discuss the reasons causing the difference.

Multiview Data

As a proof of concept, we consider the easier case of evolving a complete 3D model before testing on single-stream sequences. For this purpose we run our deformation framework on the MIT multiview mesh dataset [229], as done by Zollhöfer *et al*. [261]. It contains several sequences of 150-200 meshes, fused from multiview captures around people who are executing movements with considerably large deformation. Hence it also permits quantitative evaluation.

Figure 6.16 shows our reconstructions throughout the sequences, together with the alignment error indicating the deviation from the ground truth. We started with an SDF initialized from the first mesh and continuously evolve it towards the SDF corresponding to every next frame. While the error tends to slightly increase over time, the effects of drift accumulation are not severe. The model error remains below 2 mm throughout both sequences, with an

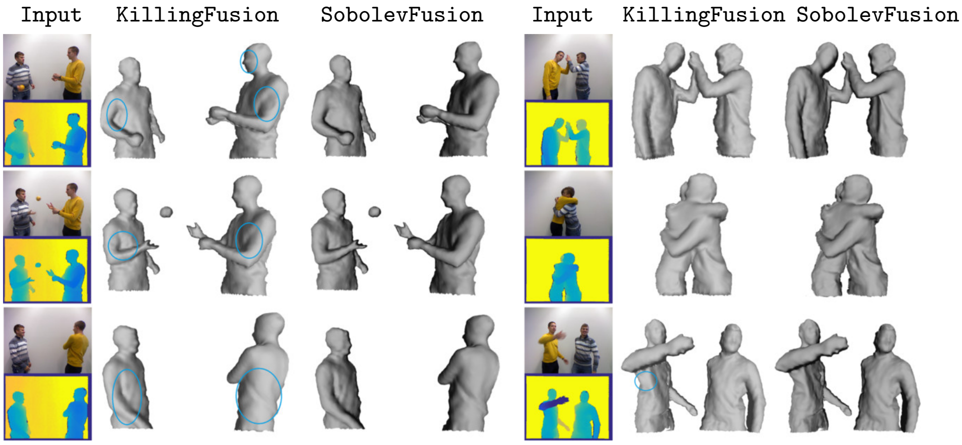


Figure 6.7: **Comparison of Sobolev pre-conditioning versus damped AKVF regularization**: SobolevFusion achieves crisper geometric details, while KillingFusion is slightly faster.

average of 1.3 mm in *Bouncing* and 0.9 mm in *Swing*. We included one of the dancing girl sequences, as they are typically used in the literature to demonstrate problems with topology changes when the dress touches the legs [56], but observe no problem for KillingFusion. In particular, we notice no larger artifacts near the dress edge than other areas of the model. The biggest errors are, in fact, typically near the hands of the subjects. This is because the used voxel size of 8 mm does not always manage to recover fine structures like the fingers with absolute accuracy. Last but not least, we noticed that if instead we deform the first SDF to every frame, more iterations are required to converge, but the errors do not change significantly.

Topologial Changes and Fast Motion

A major advantage of our proposed formulation that stays entirely within the TSDF representation is that it can inherently handle topological changes and capture large deformations. Thus we first demonstrate these abilities.

Figures 6.1 and 6.5 each show a human turning in a complete 360° loop while undergoing topology changes, such as interacting with a balloon or splitting his hands from the hips. They have been reconstructed with the Sobolev and the AKVF regularization respectively, proving that both variants of our scheme are able to recover a complete 3D model in unconstrained motion. Similarly, Figure 6.6 displays a person taking off their hat, captured with the AKVF regularizer. This proves that both versions of our reconstruction technique handle interacting subjects.

KillingFusion versus SobolevFusion

Similarly, Figure 6.7 directly, compares live frames of the two versions of our variational formulation on a recording with interacting subjects. In addition to

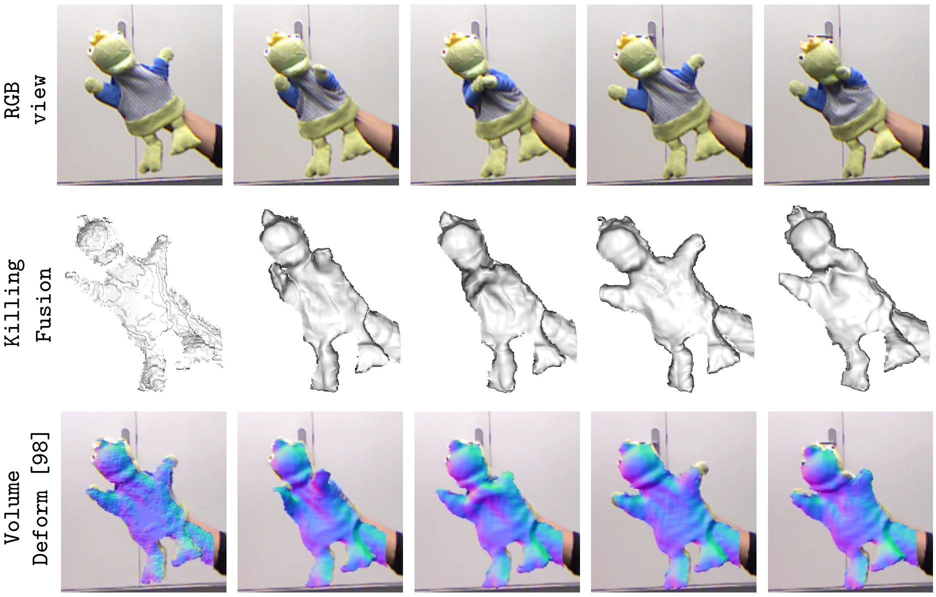


Figure 6.8: **Comparison of warped live frames on a sequence with topological changes**. Our variational approach evolves into the correct geometric shape between frames, while the correspondence-based VolumeDeform [98] is unable to track the motion when the frog hands touch.

less over-smoothing of facial features and folds on clothes, the Sobolev variant captures concavities better and defines sharper edges, both at the shape outline and where surfaces touch. However, convolving the grid with 7-voxel Sobolev kernels is more computationally demanding. Thus selecting which version of our variational formulation to use is a trade-off between speed and level of geometric detail.

Geometric Fidelity

Next, we compare our approach to a state-of-the-art non-rigid reconstruction technique that relies on correspondences, VolumeDeform [98], whose authors kindly provided their results on our recordings. In Fig. 6.8 we test on a frog puppet whose arms touch and then split again. While both VolumeDeform and our method are able to capture controlled motion, the third and fifth displayed live frames show that VolumeDeform is unable to track the hands when they touch and instead retains the canonical pose. In contrast, our approach captures this kind of motion successfully.

In addition to these qualitative observations, we carry out quantitative experiments. To be able to quantify results, we used mechanical toys that can both deform and move autonomously. We first reconstructed them in their static rest pose using a markerboard for external ground-truth pose estimation. Then we recorded their non-rigid movement sequences starting from the rest pose, which lets us evaluate the error in the final canonical-pose reconstruction.

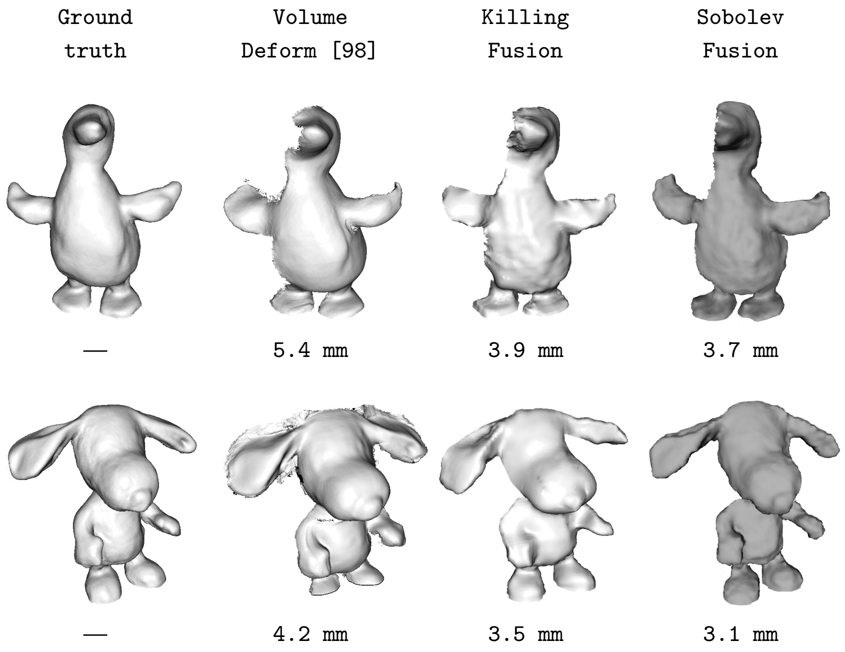


Figure 6.9: **Geometric error on objects with ground-truth canonical models from our the *Deformable 3D Reconstruction Dataset*.** Both versions of our variational formulation outperform VolumeDeform [98], as the mechanical toys in the sequences exhibit fast motion. Errors are given below the respective reconstruction.

Addressing the lack of single-stream reconstruction datasets acquired with real sensors, we make our data publicly available[1].

Figures 6.8 and 6.15 juxtapose our results with VolumeDeform [98]. Note that the reconstructions are partial because these objects do not complete 360° loops. Both approaches perform well under general motion, such as that of the *Duck* wings. However, the latter three *Snoopy* live frames show that it cannot recover once a topological change occurs when the feet touch. Furthermore, the rapid ear motion, making a full revolution from horizontal to vertical position and back within 5 frames, cannot be captured and causes artifacts in the final reconstruction, while our level-set based KillingFusion fully evolves the surface even in such cases. These results indicate that SDFs are better suited for overcoming large inter-frame motion and changing topology.

Figure 6.9 gives the *CloudCompare* model errors of the outputs of VolumeDeform [98] and both versions of our system. As our formulation is designed to handle such challenging motions, its error is lower than that of VolumeDeform. Moreover, the results show that the version with Sobolev gradient flow avoids the over-smoothing and the occurrence of spurious artifacts caused by noise that are present in the damped AKVF alternative.

[1] `http://campar.in.tum.de/personal/slavcheva/deformable-dataset/index.html`

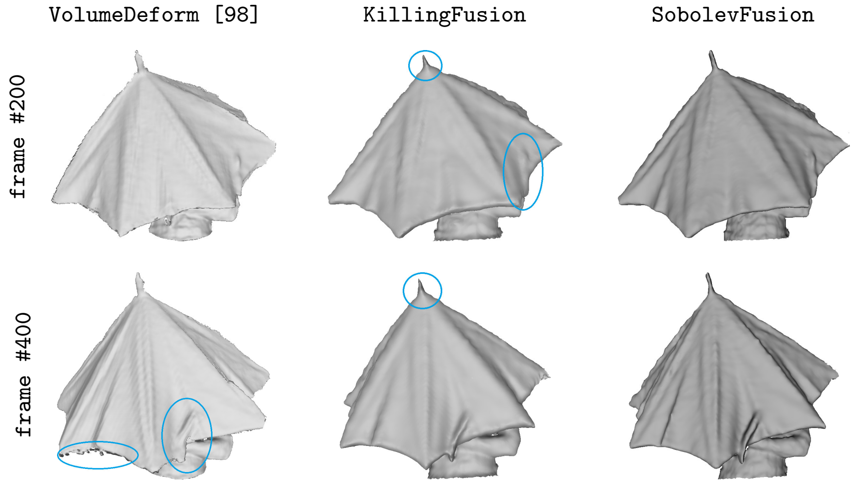


Figure 6.10: **Comparison of warped live frames of the *Umbrella* sequence from VolumeDeform** [98]. Sobolev gradient flow yields similar or higher level of detail as VolumeDeform without artifacts at the edge, while the damped AKVF deformation leads to over-smoothing of thin elements such as the tip.

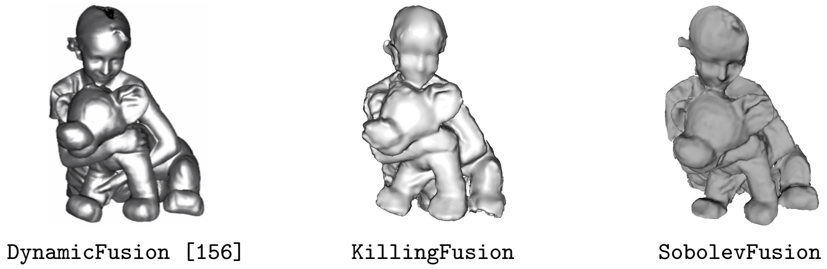


Figure 6.11: **Canonical model comparison on the full-loop *Squeeze* sequence from DynamicFusion** [156]. SobolevFusion recovers the fine structures on the face better than KillingFusion [193].

Public Data

While there are no available single-stream non-rigid reconstruction datasets with ground-truth data, some authors have made their recordings publicly available.

In Figure 6.10 we test on the *Umbrella* sequence from VolumeDeform [98]. Our method achieves a similar, or even higher, level of detail as VolumeDeform, without creating spurious elements around the edge or fusing the strap into the umbrella. Furthermore, we again observe that the Sobolev pre-conditioning scheme better captures fine structures, such as the umbrella tip, while the damped AKVF approach with level set preservation constraint tends to over-smooth such geometric details.

Next, we test on the *Boxing* sequence of VolumeDeform [98] in Figure 6.12. KillingFusion achieves similar quality. Notably, the second warped frame

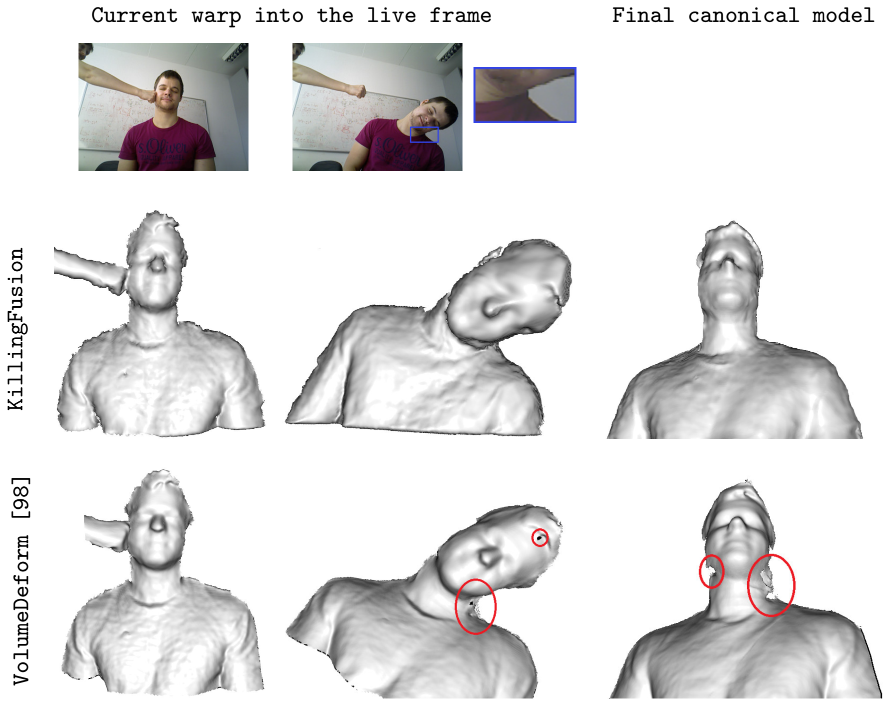


Figure 6.12: **Comparison on the *Boxing* sequence from VolumeDeform** [98]. Our depth-only KillingFusion outputs reconstructions of comparable fidelity to VolumeDeform which additionally relies on the colour frames for SIFT matching. In particular, our canonical model exhibits less artifacts where larger motion occurred, *e.g*. around the neck which bends more than 90°. Moreover, the marked regions of our live frames show that KillingFusion follows the folds of the neck more naturally.

demonstrates that our SDFs deform to the geometry more naturally: our warped model replicates the skin folding around the neck, while the model of VolumeDeform does not bend further than a certain extent, causing artifacts in the final reconstruction as well. This is similar to the behaviour we observed on our own rapid motion recordings. In conclusion, another dataset also indicates that level set evolution allows to capture larger motion better than mesh-based techniques.

We also run KillingFusion on the 360° sequences used in Dou *et al*.'s offline non-rigid bundle adjustment paper [56] and in DynamicFusion [156], namely *Andrew-Chair* in Figure 6.13 and *Squeeze* in Figure 6.11. As we do not have the authors' resulting meshes, we show snapshots available from the publications. KillingFusion manages to recover a complete model of comparable fidelity to the other techniques. In particular, despite the coarse voxel resolution, it preserves fine-scale details such as noses, ears and folds on shirts after a full loop around the subject. Moreover, we again notice that SobolevFusion captures details better, as, for instance, the facial features are much more conspicuous than for KillingFusion.

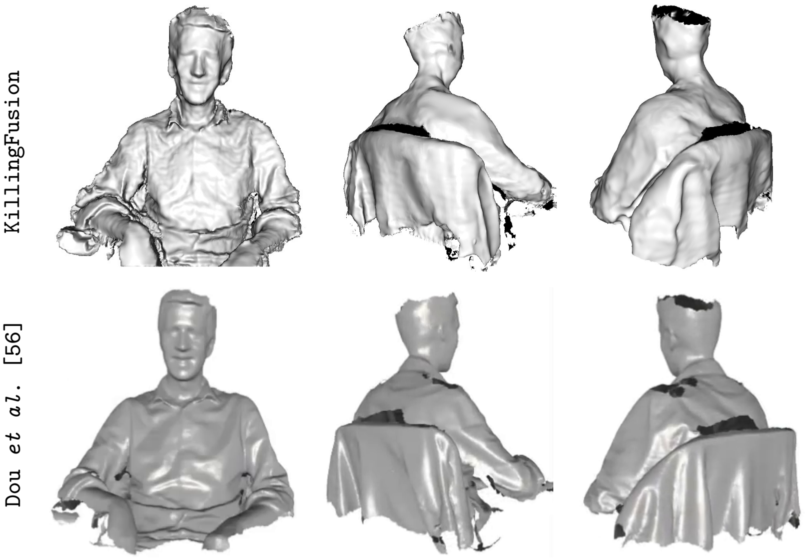


Figure 6.13: **Comparison to the offline bundle adjustment method of Dou *et al.*** [56]: our KillingFusion achieves similar quality at real time, preserving fine structures, such as shirt folds and the nose, after a full loop around the subject.

Large Motion

Even though many of the sequences used so far exhibit large motion, we simulate an extreme case of a lower frame-rate sensor by taking every n^{th} frame from 360° sequences. To this end we use the slow-motion *Andrew-Chair* from Dou *et al*. [56] and the fast *Alex* sequence, as displayed in Figure 6.14.

Both versions of our approach manage well with frequency decresed up to 5 times. Naturally, when less frames are fused, the cumulative TSDF is noisier. However, there are differences in the geometric fidelity that are not only due to noise. In particular, when only every 10^{th} frame is used, the reconstruction is still consistent for the slower *Andrew-Chair* sequence, while the faster *Alex* sequence starts creating artifacts due to misaligned geometry. Moreover, due to improved convergence of the Sobolev scheme, it manages to recover even larger motion than KillingFusion. This can be concluded from the last two columns of Figure 6.14, as the KillingFusion result for *Alex* at 10-frame speedup is similar to that of SobolevFusion for 15-frame speedup.

Last but not least, we observed that SobolevFusion requires up to 15% less iterations to converge than KillingFusion. While a single Sobolev iteration with a 7^3 kernel is slower than a single Killing iteration, the difference in numbers of iterations make the approaches comparable in terms of processing time. It is likely that KillingFusion is still slightly faster, but the ease of implementation of the SobolevFusion energy that consists of fewer terms may be more appealing.

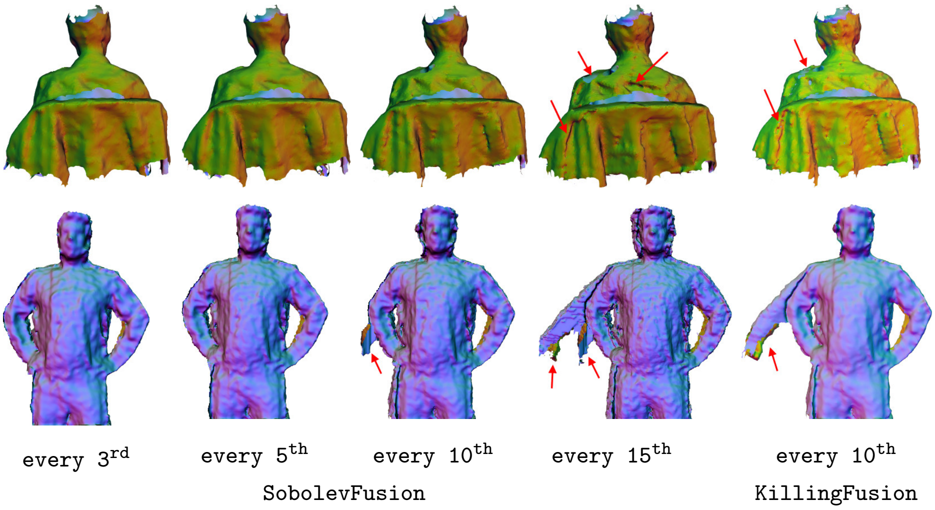


Figure 6.14: **Lower frame-rate test.** We use only every n^{th} frame, as indicated under the results. SobolevFusion outputs high-fidelity reconstructions using only 20% of the frames. For slow motion, even less frames give good results, while for large motion some of the geometry cannot be recovered, resulting in artifacts. The right-most columns show the KillingFusion result for every 10th frame, exhibiting similar degradation properties as SobolevFusion does for every 15th frame due to its better convergence.

6.5 Conclusion

We have developed a technique for non-rigid 3D reconstruction of surfaces undergoing free motion, including fast movements, changing topology and interacting subjects. Our variational energy formulation allows to determine dense deformation flow field updates without correspondence search and to avoid repeated conversion between mesh and SDF representations. Thanks to the theories of two mathematicians, Killing and Sobolev, we have proposed several regularization alternatives that ensure that a geometrically consistent reconstruction is obtained. A variety of qualitative and quantitative examples have shown that KillingFusion and SobolevFusion can recover the geometry of objects undergoing diverse kinds of deformations. Furthermore, we have contributed a quantitative evaluation dataset, hoping to shift the focus in non-rigid reconstruction from qualitative assessments, which may sometimes be misleading, to quantitative tests that put different techniques in front of the same challenges.

As the scenarios featuring fast motion, interactions and changing topology are traditionally challenging for other state-of-the-art methods, we believe that our contribution is a step forward towards making real-time capture of unconstrained motion and 3D avatar creation truly available to the general user.

However, one of the limitations of our correspondence-free scheme is that it is unable to track correspondences, similar to other methods based on the variational level set method [169]. If the objective is obtaining an accurate 3D model of the object that was captured, our technique is absolutely sufficient. Nevertheless, some applications require correspondence information, such as texture transfer, character animation or 4D video compression [44]. As we believe that SDF evolution is better suited to capturing topological changes and fast motion than correspondence-based deformation techniques, in the next chapter we set out to recover correspondences after the evolution has taken place.

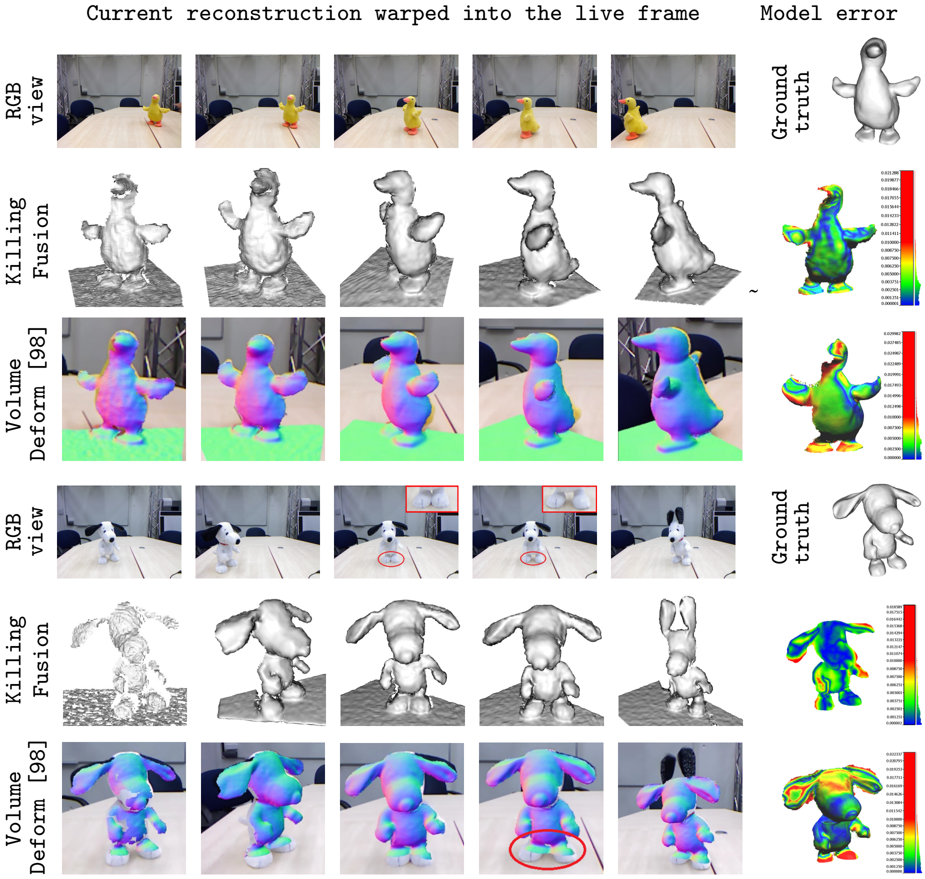


Figure 6.15: **Comparison between KillingFusion and VolumeDeform [98] under rapid motion and topological changes**. *Duck*'s wings and *Snoopy*'s ears make a complete up-down revolution within 5 frames, and *Snoopy*'s feet touch and separate several times. While a mesh-based method does not handle such motions, our SDF-based approach fully captures the deformations. This is reflected in less artifacts in the final model. Live frames are in chronological order, the objects do not complete 360° loops. Red is saturated at 1 cm in all error plots.

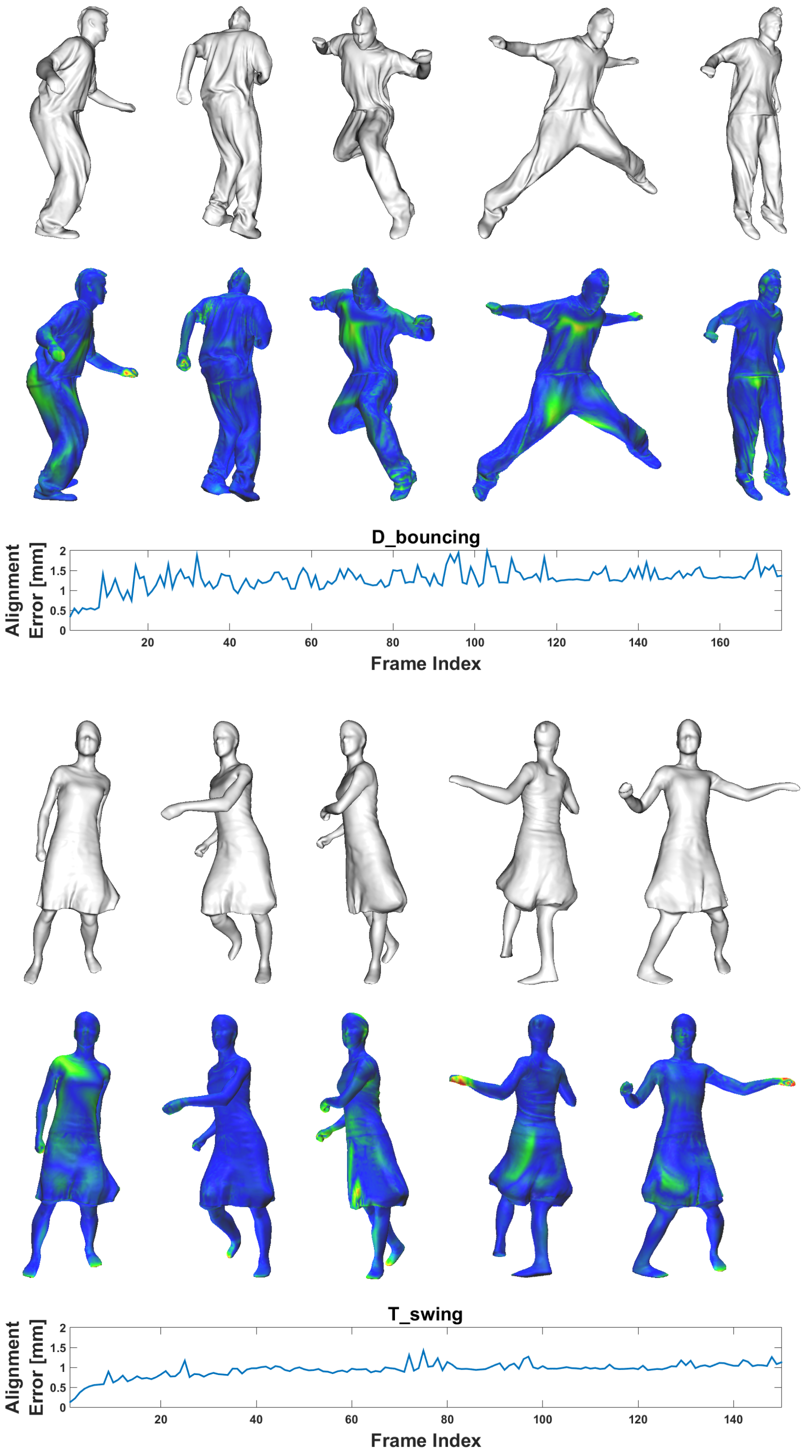


Figure 6.16: **Non-rigid registration of complete 3D shapes from the MIT dataset** [229]. Starting with an initial SDF, we gradually evolve it to match every next model in the sequence. Each pair shows our reconstruction along with its corresponding error plot, where red is saturated at 1 cm deviation.

7 Voxel Correspondence via Laplacian Eigenfunctions

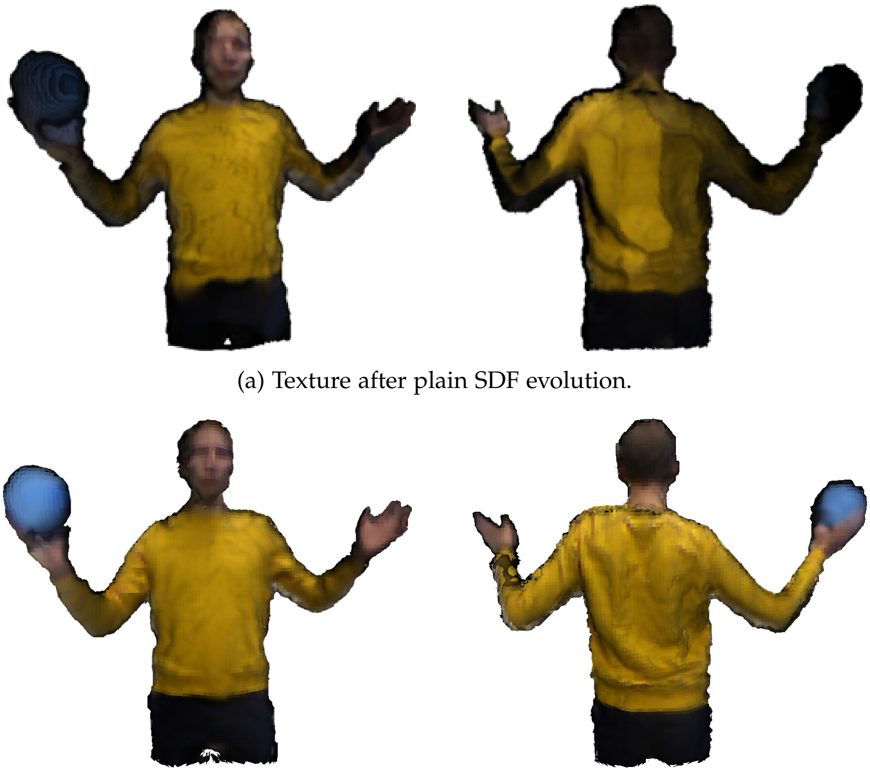
So far we have created a technique that reconstructs non-rigidly moving objects using our implicit-to-implicit energy. It applies gradient flow in order to evolve the current input towards the canonical-pose SDF and subsequently fuse its new data into it. However, the underlying variational level set method entails loss of data association, which might be needed for other applications.

To recover correspondences, we study the properties of the lowest-frequency Laplacian eigenfunctions of an SDF, as they are known to encode natural deformation patterns that the underlying shape can undergo. Therefore we will not use the implicit-to-implicit energy in this chapter, but will stay within the SDF representation and develop techniques that can be used in addition to the evolution energy.

For moderate motions we are able to obtain implicit associations via an additional data term that imposes voxel-wise eigenfunction alignment. This is not sufficient for larger motions, so we explicitly estimate voxel correspondences via signature matching of lower-dimensional embeddings of the eigenfunctions.

7.1 Introduction

Real-world scenes contain shapes that move and interact non-rigidly over time, *i.e.* they inhabit a 4D spatio-temporal domain. Multi-camera systems are able to recover complete, but independent 3D models of the scenes at isolated time instances [189]. However, these are not consistent over time as they lack motion information. Thus in order to enable tasks such as performance capture, primitives on a template 3D surface have to be tracked across frames of such motion sequences, following a deformation model. This challenging problem has numerous applications, among which virtual reality, 3D avatar animation, 4D video compression and special effects.



(a) Texture after plain SDF evolution.

(b) Texture after voxel matching.

Figure 7.1: **Comparison of texture obtained after non-rigid SDF evolution**: (a) colours would diffuse into each other if evolved with the same warp field as the SDFs, but (b) become consistent if Laplacian eigenfunction signatures are matched for voxel correspondence.

One major difficulty is capturing non-rigid motion involving topological changes, *e.g*. when subjects interact, or when loose clothing touches or splits from other surfaces. While triangular meshes have become a common discrete surface representation for motion capture, they require tedious handling for such situations [250].

On the other hand, level set methods [163, 164] inherently manage changing topology without need for additional processing. We have experienced this in the previous chapter, where multiple examples show that KillingFusion and SobolevFusion cope with such motion seamlessly. Variants of the variational level set framework are widely used in shape analysis due to the ease they provide for calculating geometric properties, such as derivatives, normals and curvature, over a fixed Cartesian grid without parameterization. However, the underlying level set evolution involves an incremental iterative numerical scheme, in which correspondences are lost [169, 242], as can be seen by the colour diffusion in Figure 7.1a. This limits applications to reconstruction and modelling, but prevents tasks that require tracking data associated with the surface, such as texture mapping and identity transfer.

To remedy this discrepancy, researchers have investigated hybrid structures combining the advantages of both meshes and level sets. For instance, SpringLS [139, 140] provide interoperability between the two representations, allowing the user to interpret geometry in the form that is more beneficial at the current step of an algorithm.

Other authors adhere to the mesh representation and use spectral methods based on the Laplace-Beltrami operator to calculate volumetric descriptors, which are matched to identify corresponding interest points across shapes. These include the volumetric heat kernel signature [172] and its scale-invariant follow-up versions [135].

Other approaches favour the level set framework. The Particle Level Set [63] and the Marker Level Set [149, 150] methods apply the estimated motion not only to the volumetric grid, but also to a set of particles attached to the surface, and subsequently correct for their locations. Similarly, Pons *et al.* [169] maintain explicit backward correspondences to the reference shape and advect them using a system of coupled Eulerian partial differential equations.

While some of these techniques demonstrate successful results on synthetic examples or in scenarios where the level set equations are analytically defined, they all entail some overhead for representation conversion, descriptor matching, or additional equation handling. To the best of our knowledge, no method has managed to integrate correspondence tracking within the level set equation itself. This is largely due to the conflicting objectives of an evolving level set energy versus direct explicit correspondence matching.

Our objective in this chapter is to develop techniques which allows to *propagate volumetric correspondences together with or after variational SDF evolution*. We propose to utilize the lowest-frequency eigenfunctions of the Laplacian matrices of the TSDFs, as they encode the inherent deformation patterns of the shapes. First, we search for *implicit correspondences* via an *eigencolour* data term that aligns these representations [194]. As it is robust only up to moderate movements, we suggest an *explicit correspondence* alternative, in which we match signatures of lower-dimensional embeddings of the eigenfunctions [195].

While this strategy for posterior correspondence estimation is antithetical to traditional approaches, which use data association in order to perform the non-rigid warping, we reckon that it is the most suitable way to incorporate correspondence into the SDF evolution scheme. As it inherently handles topological changes, which occur whenever objects interact, it paves the way towards capture of arbitrary everyday scenes.

7.2 Related Work

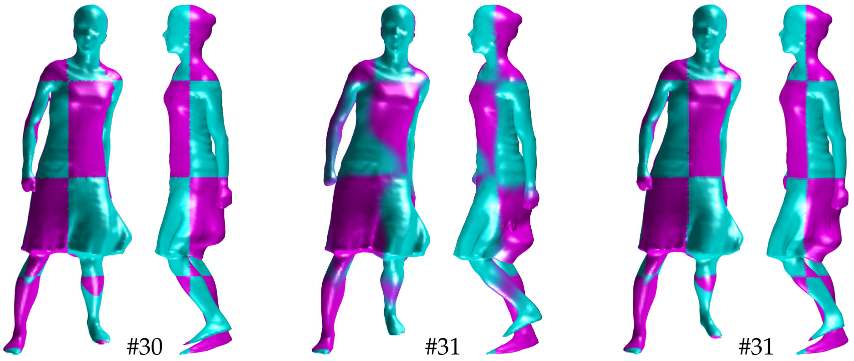
Deformable models are commonly used for 3D reconstruction, registration, simulation, animation and motion tracking [140]. Meshes and level sets are the two representations that are most often employed for manipulating the 3D data at hand. Each has advantages in certain applications, *e.g.* meshes are more suitable for registration, where correspondences between vertices need to be estimated [20, 21, 217], while level sets are more often utilized in segmentation, where the boundary of a particular structure is determined by a propagating front [163]. However, for tasks which need to combine the two representations, there either has to be an explicit structure that can be cast into either a mesh or a level set [140], or the solution needs to implicitly provide the data associated with the other representation.

Spectral descriptors Spectral decomposition methods based on the Laplace-Beltrami operator on meshes have achieved remarkable results for non-rigid full and partial shape matching [24, 133, 134, 175, 210]. They model deformations as approximate isometries of the object boundary, *i.e.* its surface. Inspired by this success, researchers have looked into volume isometries, which are more natural to be preserved during motion. This brought about the volumetric heat kernel signatures [172], volumetric maximally stable extremal regions [135], and a variety of other signatures based on Laplace-Beltrami eigendecomposition [173, 181, 182]. They typically take an arbitrarily big subset of the operator eigenfunctions in order to build a descriptor. Subsequently, quantization and matching are required in order to determine corresponding regions for the applications of non-rigid shape retrieval and classification. However, examples are limited to mainly synthetic noise-free meshes.

Hybrid structures While the spectral descriptors are computed from a mesh representation of the shape, some authors avoid it due to the difficulty of discretizing equations on polygonal grids and the tedious calculation of projections onto the discretized surface for handling properties such as gradients [14]. Instead, they prefer to use the level set framework [164, 190, 223]. It, however, does not preserve correspondences and is therefore ill-suited for tasks such as surface registration and motion tracking. Pons *et al.* [169] were among the first to propose a way to maintain correspondences during the level set evolution. They use a system of coupled PDEs in order to track backward correspondences to the initial surface position. Their framework handles large deformations and topological changes, but is based on analytically defined motion equations of curvature-dependent speed.

The Particle Level Set method [63] provides a similar scheme, in which a set of particles are associated with the initial surface. They are advected together with the level set evolution, and then processed for addition and deletion where topological changes occurred. Moreover, at each iteration, a correction step has to be done to ensure that the particles are still aligned with the surface. This is a complicated procedure, which might take hundreds of seconds. Therefore, speedups and modifications followed, such as the Marker Level Set [149, 150], which is still far from interactive frame rates.

More recently, SpringLS have been proposed to offer direct interoperability between meshes and level sets [139, 140]. They define a level set as a constellation of triangular surface elements, which are loosely connected via structures with the physical properties of springs, so that rigidity constraints can be applied. However, the processing speed still remains at the order of several minutes, while accuracy is only slightly better than that of Pons *et al.* [169].



(a) Reference. (b) Without E_{eig}. (c) With E_{eig}.

Figure 7.2: **Texture transfer from frame #30 to frame #31 of the *Swing* sequence of the *MIT dataset*** [229]: (a) reference texture; (b) colour propagated with E_{def}, showing diffusion around moving parts; (c) colour propagated with E_{def} combined with the *eigencolouring* term E_{eig}.

Voxel correspondence As the graph Laplacian of a shape is invariant to isometric deformations [128, 174], correspondence can be estimated after warping. The approach of Mateus *et al.* [147] matches voxel sets by comparing Laplacian eigenfunction signatures and reducing the problem to rigid alignment in a lower-dimensional embedded space. We modify the technique to handle SDFs of partial shapes, so that it can be used in non-rigid fusion.

7.3 Laplacian Eigencolourings

We take inspiration in part by the Laplace-Beltrami operator, whose spectrum is an isometry invariant of the shape, independent of its spatial position or parameterization, and is even dubbed to "understand" geometry [128, 174]. In analogy to physical vibration models, it is indicative of the trajectories in which a surface is able to deform [128]. The Laplace-Beltrami is an operator associated with the surface, *i.e.* the volume boundary of an object, and therefore convenient methods for calculating it from a mesh representation exist. While it is invariant to isometric deformations [182], it is more natural for the volume to be preserved during articulated motion. However, the volumetric Laplacian shares similar invariance properties only if a very fine grid with appropriate boundary conditions is used, which might be prohibitively expensive for practical 3D scenarios [182]. Nevertheless, it has been shown that the Laplacian of a voxel representation of a shape is able to handle its articulations [147].

Therefore, we propose to stay within the level set framework, where objects are represented via voxel grids. We utilize the lower-frequency eigenfunctions of the Laplacian, corresponding to its smallest eigenvalues, as they represent the base shape (*e.g.* a human body) and capture information about its natural non-rigid motion patterns, while the higher-frequency ones account for details (limbs, wrinkles) [128, 174]. This is visualized in Figure 7.9, where the lower-

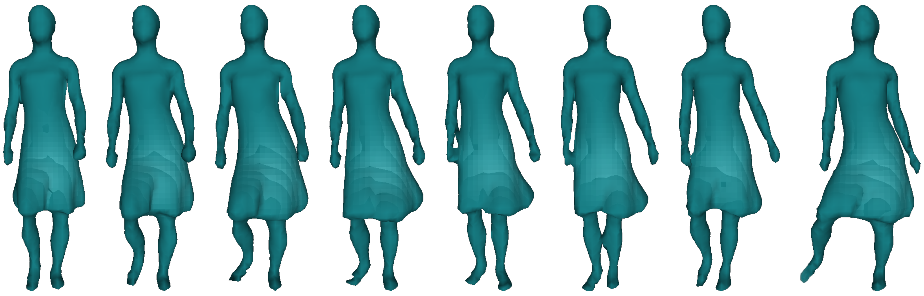


Figure 7.3: **Lowest-frequency Θ^1-eigencolourings of several poses of the same subject**. The contours form similar patterns in all cases and saturate around the skirt folds, which is the most motile region.

frequency eigenfunctions form patterns around the most motile body parts, while the higher-frequency eigenfunctions appear almost as noise. We include the eigenfunctions directly as an energy term in a variational framework. Thus, without explicitly tracking correspondences, we are able to implicitly infer them. This is demonstrated through texture transfer during level set evolution, as shown in Figure 7.2 and the results that follow: if we store an RGB grid containing the colour of each voxel and warp it in the same way as the SDF, colours would diffuse into each other, while more carefully tailored techniques result in consistent textures.

As the eigenfunction representation results in a colouring of the voxels, which describe the natural deformation modes of the shape, we also call it *eigencolouring*. To build it we first calculate the normalized graph Laplacian of the respective voxel grid. Let the number of voxels in the narrow band that is not truncated to ±1 be l - we refer to them as occupied in the current context. This is the main difference to other spectral methods, which typically consider the entire shape. The adjacency matrix W of size $l \times l$ has an entry 1 when adjacent voxels are occupied, and 0 elsewhere. Note that the diagonal entries are 0, as a voxel is not adjacent to itself. The degree matrix D contains the degree of each voxel, *i.e.* the row-wise sums of elements in W, on its diagonal. Then the normalized Laplacian is [147]:

$$L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}\,. \tag{7.1}$$

Next, we calculate its eigendecomposition $L = U\Lambda U^\top$. The full spectrum of the Laplacian (or rather, the Laplace-Beltrami) reflects all possible ways in which the shape can deform isometrically. However, since real-world data contains noise, we discard high-frequency eigenfunctions. Instead, we want to capture only the most significant characteristics of the shape, so we retain only the $K \leq 20$ eigenfunctions with smallest non-zero eigenvalues [147]. Thus we obtain the matrix U^K, which is a lower-dimensional embedding of the shape, whose columns are the K retained eigenvectors, while its l rows are the K-dimensional coordinates of the embedded shape.

As each eigenfunction is an l-element vector, we pad it to the size of the original TSDF and de-linearize its indices, obtaining Θ^e which is the *eigencolouring* of the volume for its e^{th} smallest non-zero eigenvalue. It is a scalar field of the same resolution as the TSDF and if mapped to colour values gives a colour pattern distinctive for the shape, as shown in Figure 7.3. We pad with the smallest entry of the eigenfunction so that the gradient is not reversed. Furthermore, we normalize the values to the interval $[-1; 1]$ similar to a TSDF.

Given two TSDFs which we want to align, ϕ_{input} and ϕ_{target}, we expect their K lowest-frequency eigencolourings to be similar, since they stem from the same shape in potentially different poses. However, there is no guarantee that the eigenvalues are reliably ordered in the two embeddings, so we need to determine a $K \times K$ permutation matrix P that aligns the eigenspaces of our two shapes. In addition, due to sign ambiguity, we have to determine a sign matrix M, resulting in an overall transformation $T = MP$.

In case we use only $K = 1$ eigenfunction, it always corresponds to the smallest non-trivial eigenvalue, so there is no ambiguity. For larger K, we determine the transformation T as explained in Section 7.4 and re-order the embeddings respectively. Finally, we integrate the **Laplacian eigencolourings term** into our variational formulation:

$$E_{eig}(\Psi) = \frac{1}{2} \sum_{x,y,z} \sum_{t=1}^{K} \left(\Theta_{input}(x+u, y+v, z+w) - \Theta_{target}(x,y,z)\right)^2 . \qquad (7.2)$$

The complete non-rigid evolution energy then becomes:

$$E_{def2}(\Psi) = E_{data}(\Psi) + w_{eig} E_{eig}(\Psi) + w_{reg} E_{reg}(\Psi) . \qquad (7.3)$$

As we view the eigencolourings term as another data term, we use $w_{eig} = 1$ in our experiments.

Figuresc 7.7 and 7.8 show colour transfer using correspondences estimated implicitly using E_{def2}. They demonstrate that the energy is robust for moderate motion such as a squat, but cannot handle larger deformations such as the turning dancing girl. Thus we turn to explicit voxel matching next.

7.4 Voxel Matching

The transformation T discussed in the previous section relates the reduced embeddings of the two shapes as follows:

$$(U^K_{input})^\top = T (U^K_{target})^\top . \qquad (7.4)$$

To calculate it, we seek an optimal assignment between their column eigenvectors $\mathbf{u}^i_{target}$ and $\mathbf{u}^j_{input}$, $i, j \in \{1, ..., K\}$. The approach of Mateus *et al.* [147] suggests to construct histograms from these eigenvectors, since they are invariant to the value ordering and the number of entries l, and consider them as signatures of the eigenfunctions. We thus build a 200-bin histogram $hist(\cdot)$ from each vector and store the similarity of each eigenvector pair as the ℓ_1 histogram difference in a score matrix A:

$$A_{i,j} = \min(||hist(\mathbf{u}^i_{target}) - hist(\pm \mathbf{u}^j_{input})||_1) . \qquad (7.5)$$

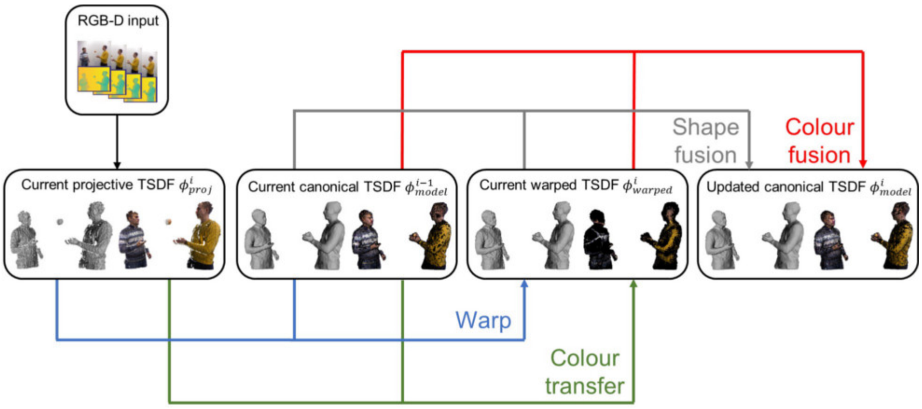

RGB-D input
Shape fusion
Colour fusion
Current projective TSDF ϕ^i_{proj}
Current canonical TSDF ϕ^{i-1}_{model}
Current warped TSDF ϕ^i_{warped}
Updated canonical TSDF ϕ^i_{model}
Warp
Colour transfer

Figure 7.4: **Complete non-rigid fusion pipeline**. First we generate the projective TSDF ϕ^i_{proj} of an input RGB-D pair from the current camera pose estimate. Then we warp it towards the current canonical model TSDF ϕ^{i-1}_{model} using our variational minimization scheme, obtaining ϕ^i_{warped}. Next, we estimate voxel correspondences between ϕ^i_{proj} and ϕ^i_{warped} in order to transfer colour to the warped TSDF. Afterwards we fuse ϕ^i_{warped} into the canonical model, obtaining its updated state ϕ^i_{model}. Finally, we run a backward warp from ϕ^i_{model} to ϕ^i_{proj} to visualize the live frame to the user.

Additionally, a matrix M' stores the sign of $\pm\mathbf{u}^j_{input}$ that yielded the lower score.

This is an assignment problem between eigenfunction signatures, which we solve for the lowest cost via the Munkres algorithm [72] over A. We then build the permutation matrix P according to its output, and look up M' for the appropriate sign in M. We thus obtain the sought transformation matrix $T = MP$ and use it to estimate correspondence, since according to Umeyama's theorem, it can be found through alignment of the two Laplacian eigenspaces [225]. The correspondences between the embeddings are transferred to the voxels of the original shapes via nearest neighbour search between the embedded- and voxel-coordinates. If a near-surface voxel is assigned to an off-surface voxel, we discard the match.

After obtaining initial matches, we use the Weiszfeld algorithm [235] to determine the geometric median in a $3 \times 3 \times 3$ neighbourhood in order to retain only the most likely correspondence. This step is crucial as we are dealing with partial TSDFs, whose Laplacian eigenfunctions might carry information about non-overlapping regions.

Implementation

We use the described strategy to transfer colour from an initial projective TSDF to its warped counterpart. In this way we are able to obtain a reliably coloured cumulative model following the complete pipeline described in Figure 7.4.

As parallelization of the voxel matching procedure is not straightforward,

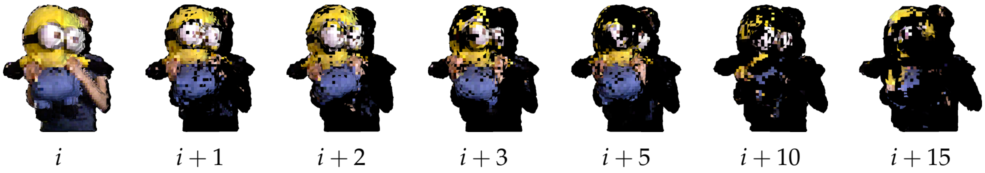


Figure 7.5: **Colour transfer from reference frame i to target frame $i+n$.** With increasing distance the amount of transferred colour decreases, but remains correct thanks to our robust voxel correspondence scheme.

in practice we run it on the CPU while the next frame(s) are being warped on the GPU. Depending on volume size, it takes 58-500 ms per frame on a 2.80 GHz Intel Core i7 CPU. When done, it continues with the latest warped frame, effectively avoiding temporal overhead.

7.5 Evaluation

To evaluate the ability of our system to determine correspondences, we look at texture transfer. If voxel matches are accurately determined, colours will not diffuse into each other over time.

First, we assess the amount of colour that can be transferred depending on the difference in pose. To this end we test on the richly textured *Minion* sequence from VolumeDeform [98]. Figure 7.5 shows the results when transferring colour from frame i to the next one, as well as to frames separated by a larger distance. The amount of texture that is being recovered decreases with the increasing pose difference, but our scheme manages to determine stable matches even when views are 15 frames apart. Furthermore, our procedure for match rejection makes sure that only reliable correspondences are returned, and thus there is no transfer of incorrect colours.

Fig. 7.1 demonstrates results on a full 360° loop sequence that was used in the previous chapter as well. When the RGB values are propagated with the same warp field as the evolving TSDF, the colours on the resulting model diffuse into each other during the interpolation process. In particular, since there is no guarantee that surface voxels remain on the surface during evolution, colours mix not only with their neighbouring ones, but also with the colour-less off-surface voxels, resulting in the observed smoky effect. One possibility to counteract this problem is to propagate colours along the normal direction, but the issue of colour diffusion will still persist.

On the other hand, our voxel matching scheme is able to recover a much clearer texture. Colours on the front are rather crisp, since the difference between the canonical pose and the initial frames is not too large and thus matching is very exact. The back shows more mixed colours, as the poses become more distant and matching becomes more challenging, but the result remains visually pleasing.

Note that our proposed technique is a first solution to combine explicit correspondence information with level set evolution. Thus the main objective

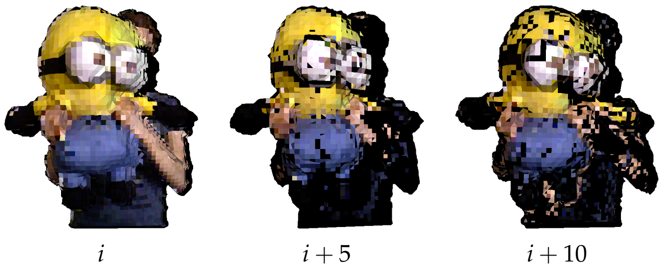


Figure 7.6: **Densifying voxel correspondence based on Laplacian eigenfunction signature matching** via expectation-maximization [147].

has been to reliably colour the reconstructions, rather than to estimate a dense set of correspondences. Nevertheless, we carry out quantitative evaluation on the *yt* sequence with Vicon markers used in BodyFusion [247], which features a human in motion.

We observed that our matching procedure typically returns a low error for markers on the torso of the subject, which is a region where mesh-based correspondences often suffer from sliding. However, since the lower-frequency Laplacian eigenfunctions do not always capture limbs, it is often not possible to find correspondences for markers located on the arms. As 12 out of the 18 Vicon markers are placed on the subject's arms, this dataset is not optimally suited for our method, which on average returns matches for half the markers per frame. Yet, our mean ℓ_1 error of 7.7 cm over the entire sequence is comparable to that of other single-stream methods that do not employ priors, namely 4.4 cm for DynamicFusion [156] and 3.7 cm for VolumeDeform [98]. A reason for the bigger error is that our method accumulates a higher discretizaiton error, since it always stays in voxel space, while others explicitly determine correspondences for deformation field calculation. Further, Table 1 of BodyFusion [247] allows us to compare the ratios of maximum to average error on the Vicon dataset: 2.0 for BodyFusion, 2.9 for DynamicFusion, 2.4 for VolumeDeform and 2.2 for our approach. This means that for DynamicFusion the maximum error deviates most from the mean, while the error of the skeleton-based BodyFusion stays most uniform throughout the sequence. Our ratio is outperformed only by that of BodyFusion, *i.e.* our algorithm is consistent over all frames and is independent of the amount of motion.

Finally, we devise another quantitative test for voxel correspondences, which allows us to test on locations that are not on limbs. For this purpose we detect SIFT features [138] on well-textured sequences, such as the *Minion* from VolumeDeform [98]. Next, we match them across frames using a very strict outlier rejection policy, so that only very accurate matches are retained. On average we kept 26 SIFT matches per frame pair. Then we carried out our voxel matching scheme as before and compared the 3D locations of the found correspondences to the back-projected SIFT keypoints, obtaining an average ℓ_1 error of 7.2 cm. Since this result is close to that on the Vicon dataset, it confirms the performance of our system. This is a promising result for the incorporation of explicit correspondences into implicit level set frameworks.

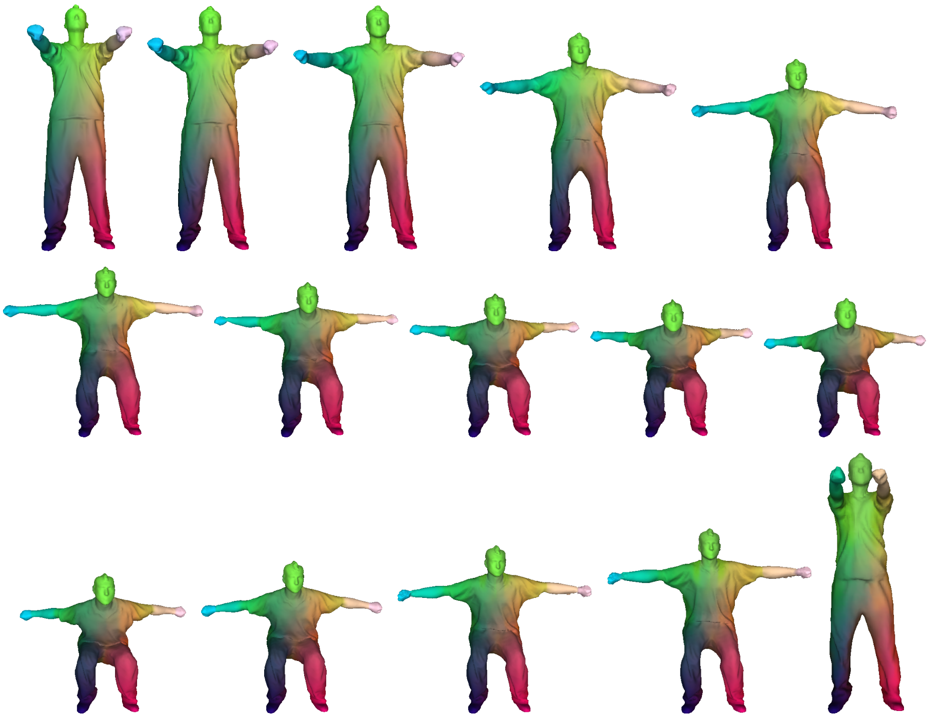


Figure 7.7: **Texture transfer via the implicit correspondence energy on the *Squat* sequence of the *MIT dataset*** [229]. When there is no abrupt motion, E_{eig} is sufficient to preserve a stable texture.

7.6 Conclusion

We have devised two voxel correspondence estimation strategies over SDFs of partial shapes, allowing realistic colouring of the obtained models when used in one of our variational SDF evolution non-rigid reconstruction schemes. Our voxel correspondence techniques allow us to stay within the SDF representation by considering the Laplacian of the shape represented in the narrow band of the voxel grid. We rely on the lowest-frequency Laplacian eigenfunctions, as they encode information about the natural deformation patterns, and consequently the non-rigid isometries, of the underlying shape. We have demonstrated the ability of the resulting methods to reduce colour diffusion and preserve texture during level set evolution, while keeping geometric accuracy at the same order of magnitude as our original techniques. These results further increase our confidence that unconstrained performance capture and 3D avatar creation under large motion will soon be achievable goals.

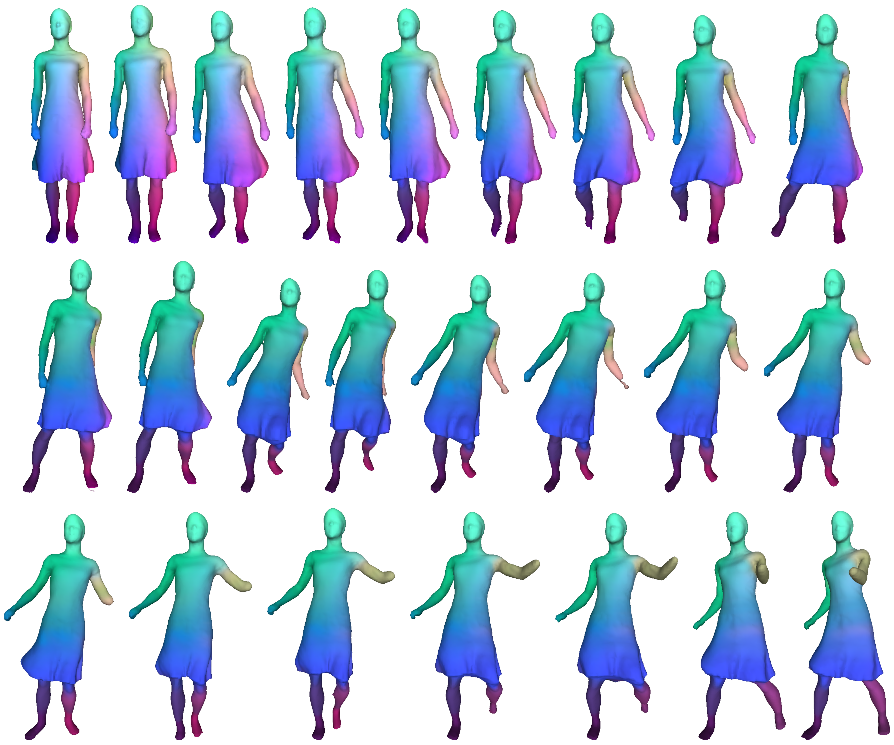


Figure 7.8: **Texture transfer via the implicit correspondence energy on the *Swing* sequence of the *MIT dataset*** [229]. Texture diffusion occurs under this larger motion as blue replaces purple on the skirt, and the geometric quality suffers as we cannot recover the arm.

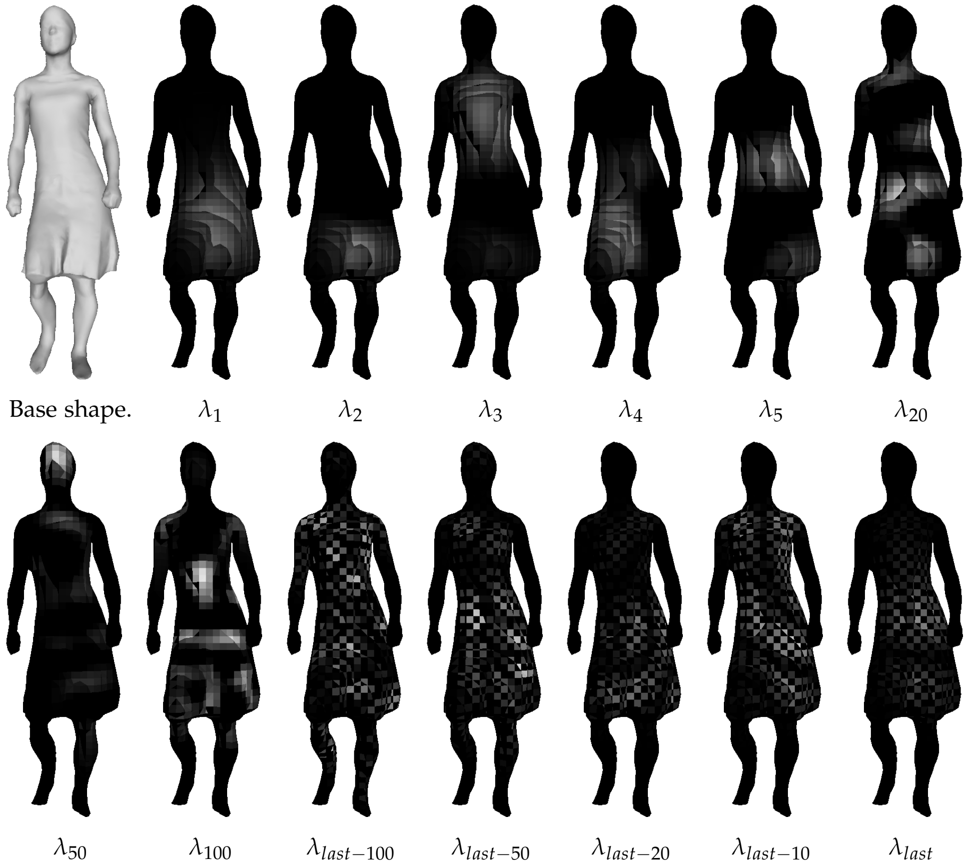


Figure 7.9: **Laplacian eigenfunction visualization**. λ_1 corresponds to the smallest eigenvalue, λ_2 to the second-smallest, *etc.*. λ_{last} denotes the largest eigenvalue, out of a total of 3151 in this case, while $\lambda_{last-10}$ is the 10^{th} largest and so on. The contours indicate that the smaller eigenvalues, corresponding to the lower-frequency eigenfunctions, capture more general and significant characteristics of the shape. On the other hand, the larger eigenvalues are associated with eigenfunctions containing a lot of high-frequency noise. Therefore, we choose the smallest eigenvalue for our framework.

Part V

Conclusion and Outlook

8 Conclusion

Here we will sum up our method and findings, analyze their advantages and limitations, and consequently propose avenues for future research.

8.1 Summary

We have developed an implicit-to-implicit correspondence-free alignment scheme between pairs of SDFs. Initially we used it for 6 DoF camera pose estimation and refinement in the context of small- to medium-scale *object reconstruction*. Then we extended this approach to *larger spaces* and inside-out SLAM-like scanning trajectories via a limited-extent volume strategy that only takes into account the most geometrically distinctive areas of the scene. These approaches lead to increased tracking accuracy and reconstructed model precision compared to other methods. Finally, we extended the technique to *non-rigidly moving objects*, where the focus was on adding rigidity constraints that make sure that the deformable motion can be factored out so that the new data can be incremented onto the canonical-pose reconstruction in a consistent manner. We proposed two strategies for this purpose: one enforces the underlying deformation field to be approximately Killing, thus generating locally isometric motions; and another one that follows gradient flow in the smoother Sobolev space, which favours global deformations rather than replicating smaller-scale details and noise. The biggest advantage of our non-rigid scheme over state-of-the-art approaches is the SDF representation used in all parts of the pipeline, which ensures that topological changes and fast motion are inherently handled without additional processing. As correspondence information is lost in the process, we proposed two strategies to recover it using the eigenfunctions of the shape Laplacian, since they are known to encode the deformation patterns that the objects can undergo. Along all of these steps we carried out extensive qualitative and quantitative evaluations, contributing new public datasets when we identified missing functionalities of existing ones.

8.2 Limitations and Future Work

While the main focus of our work has been to develop the concepts and methods that tackle open challenges in computer vision, more engineering effort has to be made should our approaches be applied at scale. For instance, while the LEV scheme ensures that camera pose estimation can be done with a low runtime and memory footprint, volumetric fusion in our approaches is still done in regular voxel grids. It should be replaced with an *efficient data structure that reduces storage requirements*, such as voxel hashing [108, 161] or a hierarchical grid [93, 109]. Another difference of these schemes to ours is that we utilize the truncated ± 1 values of the SDF in the alignment process, while others designate them as empty space and completely disregard them. Therefore a straightforward substitution of the data structures may not be sufficient, and modifications will be required to achieve the same accuracy and wide convergence basin.

A memory-efficient alternative that we have actually explored is *surfel-based fusion* in our patch-based deformable reconstruction framework described in Appendix B. Since surfels can be viewed as points with associated attributed, they are nearly as efficient as a point cloud in terms of storage. While the proposed framework achieves good results on a variety of public sequences, its quality is limited by the dependence on surfel-related parameters, while the speed is still behind real-time capabilities due to the employed expectation-maximization procedure. Nevertheless, the results are very promising, indicating that surfel-based non-rigid reconstruction is an interesting direction for future research.

Finally, the topic of recovering correspondence in the variational level set framework offers many opportunities for further exploration. As already discussed, in order to obtain *dense correspondence*, we can carry out an expectation-maximization procedure over the spectral matches. It is currently not feasible in real time, therefore possible adaptations of existing GPU-based implementations [213] have to be investigated. Alternatively, we could learn a mapping from sparse to dense fields [245], or even learn correspondences in the spectral embedding [54]. If we are to adhere to our Laplacian eigenfunction strategy, it would benefit from segmentation in the case of multiple objects, so that we can compute a separate, more representative Laplacian matrix for each one, and consequently improve the accuracy of our matches.

Non-rigid loop closure is an extremely interesting task for improving the quality and robustness of non-rigid reconstruction. It can be of tremendous help in order to mitigate errors caused by improper registration due to erratic motion, for instance. Currently mainly subject-specific approaches exist [185], but general techniques would be incredibly valuable.

8.3 Epilogue

Signed distance fields have proven to be an omnipotent tool throughout this thesis. We believe that correspondence-free approaches are extremely powerful, especially for capturing dynamic scenes that feature interactions and topology changes. Although the exciting journey taken to reach these last lines of the dissertation is now coming to an end, we believe that the developed methodology and obtained results will serve as a stepping stone for future research and, ultimately, for making precise, real-time, realistic 3D reconstruction of any scene possible.

A
Mathematical Derivations

Here we give the derivations of equations given in the main body of the dissertation. While these derivations are not essential for understanding the presented methods, we provide them for completeness.

Taylor expansion and linear system for rigid camera tracking

First we derive the Taylor expansion of E_{geom} from Eq. (4.1), which leads to the presented results in Eq. (4.5)-(4.9).

Consider the Jacobian with respect to exponential coordinates ξ:

$$\begin{aligned}\nabla_\xi \, \phi = \frac{\mathrm{d}\phi}{\mathrm{d}\xi} = \frac{\mathrm{d}\phi}{\mathrm{d}\mathbf{X}} \frac{\mathrm{d}\mathbf{X}}{\mathrm{d}\xi} = \nabla_{\mathbf{x}} \, \phi \, \begin{pmatrix} 1 & 0 & 0 & 0 & Y_3 & -Y_2 \\ 0 & 1 & 0 & -Y_3 & 0 & Y_1 \\ 0 & 0 & 1 & Y_2 & -Y_1 & 0 \end{pmatrix} = \\ = \nabla_{\mathbf{x}} \, \phi \, \begin{pmatrix} \mathbf{I}_{3\times3} & | & -\mathbf{Y}_\times \end{pmatrix} \,,\end{aligned} \tag{A.1}$$

where $\mathbf{X} = \mathbf{Y}(\xi)$ is the result of applying the transformation corresponding to ξ to the 3D point $\mathbf{Y}$, and $\mathbf{I}_{3\times3}$ is the 3×3 identity matrix.

Next, we apply first-order Taylor expansion to an SDF around the current pose estimate:

$$\begin{aligned}\phi(\xi) &= \phi\left(\xi^k\right) + \nabla_\xi \, \phi\left(\xi^k\right)\left(\xi - \xi^k\right) = \\ &= \phi\left(\xi^k\right) - \nabla_\xi \, \phi\left(\xi^k\right)\xi^k + \nabla_\xi \, \phi\left(\xi^k\right)\xi \,.\end{aligned} \tag{A.2}$$

Now we substitute the result from the Taylor expansion into the formula for E_{geom}:

$$
\begin{aligned}
E_{geom}(\xi) &= \frac{1}{2}\sum_{voxels}\left(\phi_{reference} - \phi_{current}\left(\xi\right)\right)^2 = \\
&= \frac{1}{2}\sum_{voxels}\left(\phi_{reference} - \phi_{current}\left(\xi^k\right) + \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi^k - \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi\right)^2 = \\
&= \frac{1}{2}\sum_{voxels}\Bigg(\left(\phi_{reference} - \phi_{current}\left(\xi^k\right) + \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi^k\right)^2 + \\
&\qquad + \xi^\top\,\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\,\nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi - \\
&\qquad - 2\,\left(\phi_{reference} - \phi_{current}\left(\xi^k\right) + \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi^k\right)\,\nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi\Bigg).
\end{aligned}
\tag{A.3}
$$

Next, we use the obtained expression for the derivative:

$$
\begin{aligned}
\frac{\mathrm{d}E_{geom}}{\mathrm{d}\xi} &= \frac{1}{2}\sum_{voxels}\Bigg(0 + 2\,\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\,\nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi - \\
&\qquad - 2\,\left(\phi_{reference} - \phi_{current}\left(\xi^k\right) + \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi^k\right)\,\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\Bigg) = \\
&= \sum_{voxels}\Bigg(\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\,\nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi - \\
&\qquad - \left(\phi_{reference} - \phi_{current}\left(\xi^k\right) + \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi^k\right)\,\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\Bigg).
\end{aligned}
\tag{A.4}
$$

Now we define the following matrix $\mathbf{A} \in \mathbb{R}^{6\times 6}$ and vector $\mathbf{b} \in \mathbb{R}^{6\times 1}$:

$$
\mathbf{A} = \sum_{voxels}\left(\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\,\nabla_\xi\,\phi_{current}\left(\xi^k\right)\right), \tag{A.5}
$$

$$
\mathbf{b} = \sum_{voxels}\left(\left(\phi_{reference} - \phi_{current}\left(\xi^k\right) + \nabla_\xi\,\phi_{current}\left(\xi^k\right)\,\xi^k\right)\,\nabla_\xi^\top\phi_{current}\left(\xi^k\right)\right). \tag{A.6}
$$

Finally, we use the expressions for $\mathbf{A}$ and $\mathbf{b}$ to rewrite Eq. (A.4):

$$
\begin{aligned}
\frac{\mathrm{d}E_{geom}}{\mathrm{d}\xi} &= \mathbf{A}\,\xi - \mathbf{b} \\
\Rightarrow \xi^* &= \mathbf{A}^{-1}\,\mathbf{b} \\
\xi^{k+1} &= \xi^k + \beta\,\left(\xi^* - \xi^k\right).
\end{aligned}
\tag{A.7}
$$

Derivative of the energy term for surface orientation similarity

This corresponds to deriving Eq. (4.2) into Eq. (4.11).

$$\begin{aligned}\frac{\mathrm{d}E_{norm}}{\mathrm{d}\xi_i} &= \sum_{\substack{surface\\voxels}} -\frac{\mathrm{d}\overline{\mathbf{n}}_{reference}}{\mathrm{d}\xi_i}\cdot\overline{\mathbf{n}}_{current}(\xi) - \overline{\mathbf{n}}_{reference}\cdot\frac{\mathrm{d}\overline{\mathbf{n}}_{current}(\xi)}{\mathrm{d}\xi_i} = \\ &= \sum_{\substack{surface\\voxels}} -\overline{\mathbf{n}}_{reference}\cdot\frac{\mathrm{d}\overline{\mathbf{n}}_{current}(\xi)}{\mathrm{d}\xi_i} = \\ &= \sum_{\substack{surface\\voxels}} -\overline{\mathbf{n}}_{reference}\cdot\left(\frac{\mathrm{d}\overline{\mathbf{n}}_{current}(\xi)}{\mathrm{d}\mathbf{V}}\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}\xi_i}\right) = \\ &= \sum_{\substack{surface\\voxels}} -\overline{\mathbf{n}}_{reference}\cdot(\frac{\mathrm{d}\overline{\mathbf{n}}_{current}(\xi)}{\mathrm{d}\mathbf{V}}\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}\xi}\frac{\mathrm{d}\xi}{\mathrm{d}\xi_i}) = \\ &= \sum_{\substack{surface\\voxels}} -\overline{\mathbf{n}}_{reference}\cdot\left(\nabla_{\mathbf{x}}\,\overline{\mathbf{n}}_{current}(\xi)\,\left(\mathbf{I}_{3\times 3}\quad|\quad -(\mathbf{V}(\xi^{-1}))_{\times}\right)\,\delta_i\right),\end{aligned} \tag{A.8}$$

where δ_i is a 6-element one-hot vector of zeros with ith component 1.

Gradient descent for rigid camera pose refinement

Here the frame-to-model SDF-2-SDF equation has the following form:

$$E_{global}(\xi) = \frac{1}{2}\sum_{voxels}\left(\phi_{model} - \phi_{current}(\xi)\right)^2. \tag{A.9}$$

We calculate its derivative:

$$\begin{aligned}\frac{\mathrm{d}E_{global}}{\mathrm{d}\xi} &= \frac{1}{2}\sum_{voxels} 2\left(\phi_{model} - \phi_{current}\left(\xi\right)\right)\frac{\mathrm{d}\left(-\phi_{current}(\xi)\right)}{\mathrm{d}\xi} = \\ &= \sum_{voxels}\left(\phi_{current}\left(\xi\right) - \phi_{model}\right)\frac{\mathrm{d}\phi_{current}(\xi)}{\mathrm{d}\xi} = \\ &= \sum_{voxels}\left(\phi_{current}\left(\xi\right) - \phi_{model}\right)\frac{\mathrm{d}\phi_{current}(\xi)}{\mathrm{d}\mathbf{V}}\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}\xi} = \\ &= \sum_{voxels}\left(\phi_{current}\left(\xi\right) - \phi_{model}\right)\nabla_{\xi}\phi_{cur}(\xi)\,.\end{aligned} \tag{A.10}$$

Finally, we obtain the gradient descent update for each frame $t \neq 0$:

$$\xi_t^{k+1} = \xi_t^k - \alpha\sum_{voxels}\left(\phi_{current}\left(\xi_t^k\right) - \phi_{model}\right)\nabla_{\xi}\,\phi_{current}\left(\xi_t^k\right)\,. \tag{A.11}$$

SDF evolution: Data term

The data term aligns the projective TSDF ϕ_{proj} of the current frame with the cumulative TSDF ϕ_{model}, driving their voxel-wise difference to be minimal:

$$E_{data}(\Psi) = \frac{1}{2} \sum_{x,y,z} \big(\phi_{proj}(x+u, y+v, z+w) - \phi_{model}(x,y,z)\big)^2 . \qquad \text{(A.12)}$$

$$\begin{aligned}
\frac{\partial E_{data}}{\partial u} &= \frac{1}{2}\left[\frac{\partial\big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)^2}{\partial u} - \operatorname{div}\frac{\partial\big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)^2}{\partial \nabla u}\right] = \\
&= \frac{1}{2}\frac{\partial\big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)^2}{\partial u} = \\
&= \frac{1}{2}\, 2\, \big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)\frac{\partial\big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)}{\partial u} = \\
&= \big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)\frac{\partial \phi_{proj}(x+u,y+v,z+w)}{\partial u} = \\
&= \big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)\ \nabla_x\phi_{proj}(x+u,y+v,z+w)
\end{aligned} \qquad \text{(A.13)}$$

Above $\nabla_x\phi$ denotes the x-component of the spatial gradient of the TSDF ϕ, which is obtained numerically via central differences. We will use analogous notation for the y- and z-components. The full TSDF gradient is therefore written as $\nabla\phi = (\nabla_x\phi, \nabla_y\phi, \nabla_z\phi)^\top$.

We also use the nabla symbol ∇ to denote energy derivatives. Thus:

$$\begin{aligned}
\nabla E_{data}(\Psi) &= \big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)\begin{pmatrix}\nabla_x\phi_{proj}(x+u,y+v,z+w)\\ \nabla_y\phi_{proj}(x+u,y+v,z+w)\\ \nabla_z\phi_{proj}(x+u,y+v,z+w)\end{pmatrix} = \\
&= \big(\phi_{proj}(x+u,y+v,z+w)-\phi_{model}(x,y,z)\big)\ \nabla\phi_{proj}(x+u,y+v,z+w) = \\
&= \big(\phi_{proj}(\Psi)-\phi_{model}\big)\ \nabla\phi_{proj}(\Psi)
\end{aligned} \qquad \text{(A.14)}$$

We use $\phi_{proj}(\Psi)$ to refer to the evolved TSDF after the application of the warp field vector (u, v, w), *i.e.* equivalently to $\phi_{proj}(x+u, y+v, z+w)$. We will use this shorthand notation from here onwards.

SDF evolution: Uniform motion term

The term which encourages nearby vectors to be similar is the Tikhonov-type regularizer:

$$E_{smooth}(\Psi) = \frac{1}{2} \sum_{x,y,z} \big(|\nabla U(x,y,z)|^2 + |\nabla V(x,y,z)|^2 + |\nabla W(x,y,z)|^2\big) . \qquad \text{(A.15)}$$

$$\begin{aligned}\frac{\partial E_{smooth}}{\partial u} &= \frac{1}{2}\left[\frac{\partial\big(|\nabla U(x,y,z)|^2+|\nabla V(x,y,z)|^2+|\nabla W(x,y,z)|^2\big)}{\partial u}-\right.\\ &\left.-\operatorname{div}\frac{\partial\big(|\nabla U(x,y,z)|^2+|\nabla V(x,y,z)|^2+|\nabla W(x,y,z)|^2\big)}{\partial\nabla u}\right]=\\ &=\frac{1}{2}\left[0-\operatorname{div}\frac{\partial\big(|\nabla U(x,y,z)|^2+|\nabla V(x,y,z)|^2+|\nabla W(x,y,z)|^2\big)}{\partial\nabla u}\right]=\\ &=-\frac{1}{2}\operatorname{div}\frac{\partial|\nabla U(x,y,z)|^2}{\partial\nabla u}=-\frac{1}{2}\operatorname{div}2\nabla U(x,y,z)=-\operatorname{div}\nabla U=-\Delta U\end{aligned}\tag{A.16}$$

The symbol Δ denotes the Laplacian of its operand. Thus:

$$\nabla E_{smooth}(\Psi)=-(\Delta U,\Delta V,\Delta W)^\top\tag{A.17}$$

SDF evolution: Approximately Killing vector field term

The approximately Killing vector field term (AKVF) enforces the warp field to be divergence free by minimizing the Frobenius norm of the Killing condition:

$$E_{akvf}(\Psi)=\frac{1}{2}\sum_{x,y,z}\left\|J_\Psi+J_\Psi^\top\right\|_F^2.\tag{A.18}$$

The Jacobian of the vector field is: $J_\Psi=\begin{pmatrix}\partial U/\partial x & \partial U/\partial y & \partial U/\partial z\\ \partial V/\partial x & \partial V/\partial y & \partial V/\partial z\\ \partial W/\partial x & \partial W/\partial y & \partial W/\partial z\end{pmatrix}=$ $\begin{pmatrix}U_x & U_y & U_z\\ V_x & V_y & V_z\\ W_x & W_y & W_z\end{pmatrix}$ and its transpose is denoted by $J_\Psi^\top$.

Next, let us rewrite Eq. (6.7) using the column-wise stacking operator $vec(A)$, which denotes the vectorized matrix A. Thus, $vec(J_\Psi)\in\mathbb{R}^{9\times1}$ is the 9-element vector of stacked elements from J_Ψ, and similarly $vec(J_\Psi^\top)\in\mathbb{R}^{9\times1}$ contains the elements from $J_\Psi^\top$. Finally, $vec(J_\Psi)^\top\in\mathbb{R}^{1\times9}$ denotes the transpose of $vec(J_\Psi)$.

$$vec(J_\Psi)=\begin{pmatrix}U_x & V_x & W_x & U_y & V_y & W_y & U_z & V_z & W_z\end{pmatrix}^\top\tag{A.19}$$

$$\begin{aligned} E_{akvf}(\Psi) &= \frac{1}{2}\sum_{x,y,z} \left\| \begin{pmatrix} 2U_x & V_x+U_y & W_x+U_z \\ V_x+U_y & 2V_y & W_y+V_z \\ W_x+U_z & W_y+V_z & 2W_z \end{pmatrix} \right\|_F^2 = \\ &= \frac{1}{2}\sum_{x,y,z} vec(J_\Psi + J_\Psi^\top)^\top vec(J_\Psi + J_\Psi^\top) = \\ &= \frac{1}{2}\sum_{x,y,z} \Big(vec(J_\Psi)^\top vec(J_\Psi) + 2vec(J_\Psi^\top)^\top vec(J_\Psi) + vec(J_\Psi^\top)^\top vec(J_\Psi^\top) \Big) = \\ &= \sum_{x,y,z} vec(J_\Psi)^\top vec(J_\Psi) + vec(J_\Psi^\top)^\top vec(J_\Psi) = \\ &= \sum_{x,y,z} \Big(2U_x^2 + 2V_y^2 + 2W_z^2 + U_y^2 + U_z^2 + V_x^2 + V_z^2 + W_x^2 + W_y^2 + 2V_xU_y + 2W_xU_z + 2W_yV_z \Big) \end{aligned} \tag{A.20}$$

$$\begin{aligned} \frac{\partial E_{akvf}}{\partial u} &= \frac{\partial(2U_x^2 + 2V_y^2 + 2W_z^2 + U_y^2 + U_z^2 + V_x^2 + V_z^2 + W_x^2 + W_y^2 + 2V_xU_y + 2W_xU_z + 2W_yV_z)}{\partial u} - \\ &- \frac{\partial}{\partial x}\frac{\partial(2U_x^2 + 2V_y^2 + 2W_z^2 + U_y^2 + U_z^2 + V_x^2 + V_z^2 + W_x^2 + W_y^2 + 2V_xU_y + 2W_xU_z + 2W_yV_z)}{\partial U_x} - \\ &- \frac{\partial}{\partial y}\frac{\partial(2U_x^2 + 2V_y^2 + 2W_z^2 + U_y^2 + U_z^2 + V_x^2 + V_z^2 + W_x^2 + W_y^2 + 2V_xU_y + 2W_xU_z + 2W_yV_z)}{\partial U_y} - \\ &- \frac{\partial}{\partial z}\frac{\partial(2U_x^2 + 2V_y^2 + 2W_z^2 + U_y^2 + U_z^2 + V_x^2 + V_z^2 + W_x^2 + W_y^2 + 2V_xU_y + 2W_xU_z + 2W_yV_z)}{\partial U_z} = \\ &= 0 - \frac{\partial}{\partial x}(4U_x) - \frac{\partial}{\partial y}(2U_y + 2V_x) - \frac{\partial}{\partial z}(2U_z + 2W_x) = \\ &= -4U_{xx} - (2U_{yy} + 2V_{xy}) - (2U_{zz} + 2W_{xz}) = -2(2U_{xx} + U_{yy} + U_{zz} + V_{xy} + W_{xz}) \end{aligned} \tag{A.21}$$

Similarly:

$$\begin{aligned} \frac{\partial E_{akvf}}{\partial v} &= -2(V_{xx} + 2V_{yy} + V_{zz} + U_{xy} + W_{yz}) \\ \frac{\partial E_{akvf}}{\partial w} &= -2(W_{xx} + W_{yy} + 2W_{zz} + U_{xz} + V_{yz}) \end{aligned} \tag{A.22}$$

Finally,

$$\begin{aligned} \nabla E_{akvf}(\Psi) &= -2\begin{pmatrix} 2U_{xx} + U_{yy} + U_{zz} + V_{xy} + W_{xz} \\ V_{xx} + 2V_{yy} + V_{zz} + U_{xy} + W_{yz} \\ W_{xx} + W_{yy} + 2W_{zz} + U_{xz} + V_{yz} \end{pmatrix} = \\ &= -2\begin{pmatrix} U_{xx} + U_{yy} + U_{zz} \\ V_{xx} + V_{yy} + V_{zz} \\ W_{xx} + W_{yy} + W_{zz} \end{pmatrix} - 2\begin{pmatrix} U_{xx} + V_{xy} + W_{xz} \\ U_{xy} + V_{yy} + W_{yz} \\ U_{xz} + V_{yz} + W_{zz} \end{pmatrix} = \\ &= -2\begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix} - 2\begin{pmatrix} \partial(\mathrm{div}\Psi)/\partial x \\ \partial(\mathrm{div}\Psi)/\partial y \\ \partial(\mathrm{div}\Psi)/\partial z \end{pmatrix}, \end{aligned} \tag{A.23}$$

where $\mathrm{div}\Psi = U_x + V_y + W_z$ is the divergence of the warp field Ψ.

SDF evolution: Damped Killing term

As discussed, the condition from Eq. (6.7) is too strong to account for large deformations. Re-writing the first term from the vectorized form in Eq. (A.20) sum leads to:

$$\begin{aligned}\sum_{x,y,z} vec(J_\Psi)^\top vec(J_\Psi) &= \sum_{x,y,z} \left(U_x^2 + U_y^2 + U_x^2 + V_x^2 + V_y^2 + V_z^2 + W_x^2 + W_y^2 + W_z^2\right) = \\ &= \sum_{x,y,z} \left(|\nabla U|^2 + |\nabla V|^2 + |\nabla W|^2\right) = E_{smooth}(\Psi)\end{aligned} \tag{A.24}$$

Thus increasing the weight of the motion smoothness component and decreasing the weight of the rigidity component leads to the damped Killing condition:

$$E_{Killing}(\Psi) = \sum_{x,y,z} \left(vec(J_\Psi)^\top vec(J_\Psi) + \gamma vec(J_\Psi^\top)^\top vec(J_\Psi)\right). \tag{A.25}$$

The factor γ controls the balance between the strictly rigid and non-rigid components of the regularization. A choice of $\gamma = 1$ would lead to the AKVF condition from the previous section. As we aim to alleviate the effect of the rigidity constraint, we use values of $\gamma < 1$ in our optimization. The combined functional derivative is then:

$$\nabla E_{Killing}(\Psi) = -2(\Delta U, \Delta V, \Delta W)^\top - 2\gamma \left(\frac{\partial}{\partial x}(\mathrm{div}\Psi), \frac{\partial}{\partial y}(\mathrm{div}\Psi), \frac{\partial}{\partial z}(\mathrm{div}\Psi)\right)^\top. \tag{A.26}$$

SDF evolution: Level set term

Maintaining the property of unity gradient ensures geometrically correct TSDF evolution:

$$E_{level\,set}(\Psi) = \frac{1}{2} \sum_{x,y,z} \left(|\nabla \phi_{proj}(x+u, y+v, z+w)| - 1\right)^2. \tag{A.27}$$

Note that when the implementation is over a truncated signed distance field, the gradient magnitude is unit in the narrow band and 0 in the truncated ±1 regions. If the TSDF is also scaled, the scale δ has to be applied also to the unity in the narrow band. Furthermore, values on the border between truncated and non-truncated region will be between 0 and $1/\delta$, so additional care has to be taken there.

The functional derivative is then:

$$
\begin{aligned}
\frac{\partial E_{level\,set}}{\partial u} &= \frac{1}{2}\Bigg[\frac{\partial\big(|\nabla\phi_{proj}(x+u,y+v,z+w)|-1\big)^2}{\partial u} - \operatorname{div}\frac{\partial\big(|\nabla\phi_{proj}(x+u,y+v,z+w)|-1\big)^2}{\partial\nabla u}\Bigg] = \\
&= \frac{1}{2}\frac{\partial\big(|\nabla\phi_{proj}(x+u,y+v,z+w)|-1\big)^2}{\partial u} = \\
&= \frac{1}{2}\,2\,\big(|\nabla\phi_{proj}(x+u,y+v,z+w)|-1\big)\frac{\partial\big(|\nabla\phi_{proj}(x+u,y+v,z+w)|-1\big)}{\partial u} = \\
&= \big(|\nabla\phi_{proj}(\Psi)|-1\big)\frac{\partial\Big(\big(\frac{\partial\phi_{proj}(\Psi)}{\partial x}\big)^2+\big(\frac{\partial\phi_{proj}(\Psi)}{\partial y}\big)^2+\big(\frac{\partial\phi_{proj}(\Psi)}{\partial z}\big)^2\Big)^{1/2}}{\partial u} = \\
&= \frac{|\nabla\phi_{proj}(\Psi)|-1}{2|\nabla\phi_{proj}(\Psi)|_\varepsilon}\Bigg(2\frac{\partial\phi_{proj}(\Psi)}{\partial x}\frac{\partial}{\partial u}\frac{\partial\phi_{proj}(\Psi)}{\partial x}+2\frac{\partial\phi_{proj}(\Psi)}{\partial y}\frac{\partial}{\partial u}\frac{\partial\phi_{proj}(\Psi)}{\partial y}+2\frac{\partial\phi_{proj}(\Psi)}{\partial z}\frac{\partial}{\partial u}\frac{\partial\phi_{proj}(\Psi)}{\partial z}\Bigg) = \\
&= \frac{|\nabla\phi_{proj}(\Psi)|-1}{|\nabla\phi_{proj}(\Psi)|_\varepsilon}\Big(\nabla_x\phi_{proj}(\Psi)\nabla_{xx}\phi_{proj}(\Psi)+\nabla_y\phi_{proj}(\Psi)\nabla_{xy}\phi_{proj}(\Psi)+\nabla_z\phi_{proj}(\Psi)\nabla_{xz}\phi_{proj}(\Psi)\Big) = \\
&= \frac{|\nabla\phi_{proj}(\Psi)|-1}{|\nabla\phi_{proj}(\Psi)|_\varepsilon}\big(\nabla_{xx}\phi_{proj}(\Psi)\quad\nabla_{xy}\phi_{proj}(\Psi)\quad\nabla_{xz}\phi_{proj}(\Psi)\big)\,\nabla\phi_{proj}(\Psi)\,,
\end{aligned}
\tag{A.28}
$$

where $|\cdot|_\epsilon$ denotes the norm plus a small constant ϵ which avoids division by zero. Similarly we obtain:

$$
\nabla E_{\substack{level\\set}}(\Psi) = \frac{|\nabla\phi_{proj}(\Psi)|-1}{|\nabla\phi_{proj}(\Psi)|_\varepsilon}\begin{pmatrix}\nabla_{xx}\phi_{proj}(\Psi) & \nabla_{xy}\phi_{proj}(\Psi) & \nabla_{xz}\phi_{proj}(\Psi)\\ \nabla_{yx}\phi_{proj}(\Psi) & \nabla_{yy}\phi_{proj}(\Psi) & \nabla_{yz}\phi_{proj}(\Psi)\\ \nabla_{zx}\phi_{proj}(\Psi) & \nabla_{zy}\phi_{proj}(\Psi) & \nabla_{zz}\phi_{proj}(\Psi)\end{pmatrix}\nabla\phi_{proj}(\Psi)\,,
\tag{A.29}
$$

where the 3×3 matrix in the middle is the Hessian $H_{\phi_{proj}(\Psi)}$ of the warped TSDF.

Sobolev Kernels

Here we explain how to obtain the three separable 1D filters starting with from the following equation from the paper:

$$
(Id-\lambda\Delta)S = v\,.
\tag{A.30}
$$

Let the size of the 3D Sobolev filter we are interested in be $s\times s\times s$. Then the terms in the above equation are as follows:

- Id is the $s^3\times s^3$ identity matrix.
- Δ is the s-point stencil finite difference Laplacian matrix describing neighbouring voxels, resulting in the occupancy shown in Figure A.1.
- v is a one-hot s^3-element vector with 1 at the middle index $\left\lfloor\frac{s^3}{2}\right\rfloor$ (assuming indexing starting at 0). It corresponds to a discretized Dirac impulse of size $s\times s\times s$ voxels.
- S is the s^3-element solution of the linear system that we are looking for. By restructuring it into a $s\times s\times s$ volume, we obtain the sought 3D Sobolev filter.

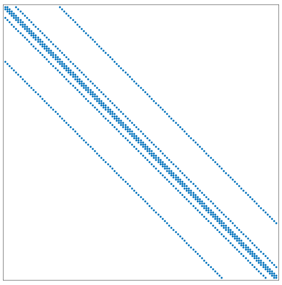


Figure A.1: Occupancy of a $s^3 \times s^3$ matrix Δ.

In order to obtain the corresponding 1D filters, we make an approximation using the higher-order SVD decomposition of the tensor S. It yields three $s \times s$ U-matrices with equal elements. We take the first singular vector from each of these matrices, obtaining the approximated 1D filters S_x, S_y and S_z. Note that they have equal entries, but we use the subscript to indicate the spatial direction in which they are applied.

This procedure needs to be done only once for selected neighbourhood size s and Sobolev parameter λ, after which the 1D filter entries can be stored. The separable convolutions are then applied over the energy derivative in each gradient descent step.

B Patch-based Non-rigid 3D Reconstruction from a Single Depth Stream

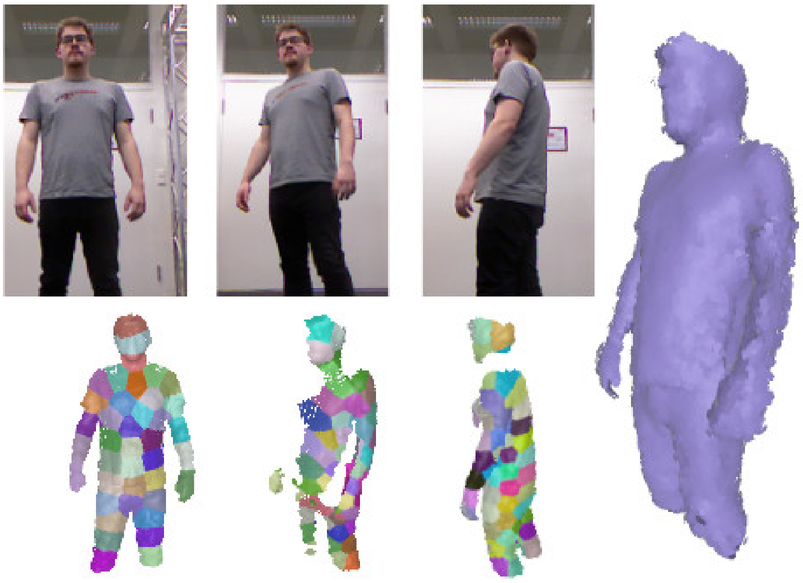


Figure B.1: **Patch-based non-rigid 3D reconstruction from a single depth stream**. Each input frame is subdivided into surface patches and deformed towards the canonical-pose model via a probabilistic non-rigid deformation framework. It imposes rigidity constraints by assuming that each patch is rigid and is connected non-rigidly to its neighbouring patches. This strategy provides robustness to occlusions and noisy data, leading to a geometrically consistent 3D model of the deforming object, as shown on the right.

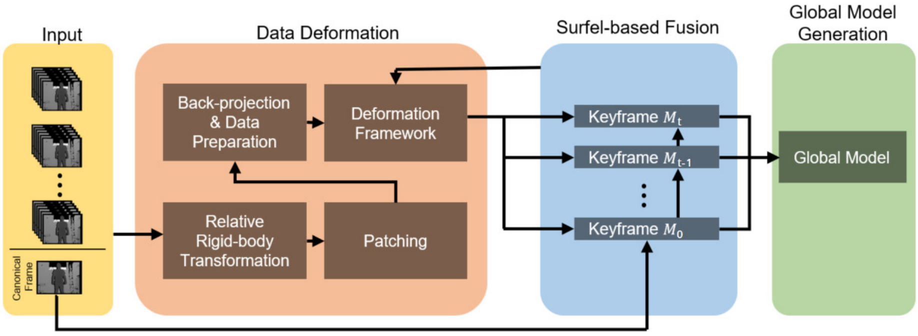


Figure B.2: **Patch-based non-rigid reconstruction pipeline**. We split the input sequence into subsequences of frames, called keyframes, in each of which a local model is built following the proposed expectation-maximization non-rigid patch-based deformation framework. The final output is obtained via a global fusion approach, which propagates and concatenates correspondences through keyframes and traces them back to the canonical pose.

In this project [118] we develop an approach for 3D reconstruction and tracking of dynamic surfaces captured with a single RGB-D sensor. It is robust to rapid motions, noisy data and occlusions due to the underlying probabilistic expectation-maximization non-rigid registration framework. Our pipeline subdivides each input depth image into non-rigidly connected rigid surface patches, and deforms it towards the canonical pose by estimating a 6 DoF transformation for each patch. The powerful combination of a data term imposing similarity between model and data, and a regularizer enforcing as-rigid-as-possible motion of neighbouring patches ensures that we can handle large deformations, while coping with sensor noise. In contrast to most existing techniques that require repeated conversion between mesh and SDF representation, we employ a surfel-based fusion technique. Last but not least, a robust keyframe-based scheme allows us to keep track of correspondences throughout the entire sequence.

The advantage of this technique over the implicit-to-implicit scheme developed in this dissertation is the direct availability of correspondences throughout the entire sequence. One of its main differences to existing approaches is the accumulation of recovered geometry in a surfel representation rather than an SDF. As surfels do not have explicit connectivity like a mesh, we expect that the approach is better-suited to handling topological changes than correspondence-based warp field techniques like DynamicFusion [156] and VolumeDeform [98]. While this has proven true in some examples, the patch-based framework still does not outperform KillingFusion and SobolevFusion in terms of changing topology. We suspect that this is due to the sensitivity of the surfel-based fusion to its parameter settings [112] and believe it can be improved in future work.

C
Authored and Co-authored Publications

Authored:

1. Slavcheva, M., Kehl, W., Navab, N., Ilic, S.: SDF-2-SDF: Highly Accurate 3D Object Reconstruction. In: European Conference on Computer Vision (ECCV) (2016)

2. Slavcheva, M., Ilic, S.: SDF-TAR: Parallel Tracking and Refinement in RGB-D Data using Volumetric Registration. In: British Machine Vision Conference (BMVC) (2016)

3. Slavcheva, M., Kehl, W., Navab, N., Ilic, S.: SDF-2-SDF Registration for Real-time 3D Reconstruction from RGB-D Data. International Journal of Computer Vision (IJCV) **126**(6), 615–636 (2017)

4. Slavcheva, M., Baust, M., Cremers, D., Ilic, S.: KillingFusion: Non-rigid 3D Reconstruction without Correspondences. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
   – *Spotlight presentation*

5. Slavcheva, M., Baust, M., Ilic, S.: Towards Implicit Correspondence in Signed Distance Field Evolution. In: PeopleCap Workshop, IEEE International Conference on Computer Vision (ICCVW) (2017)
   – *Best workshop paper award*

6. Slavcheva, M., Baust, M., Ilic, S.: SobolevFusion: 3D Reconstruction of Scenes Undergoing Free Non-rigid Motion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
   – *Spotlight presentation*

7. Slavcheva, M., Baust, M., Ilic, S.: Variational Level Set Evolution for Non-rigid 3D Reconstruction from a Single Depth Camera. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) (2018)
   – *Submitted*

Co-authored:

1. Kozlov, C., Slavcheva, M., Ilic, S.: Patch-based Non-rigid 3D Reconstruction from a Single Depth Stream. In: International Conference on 3D Vision (3DV) (2018)

Bibliography

[1] KinectFusion Implementation in the Point Cloud Library (PCL). `https://github.com/PointCloudLibrary/pcl/tree/master/gpu/kinfu`. Last accessed: November 6, 2015

[2] Abhijit, J.: Kinect for Windows SDK Programming Guide. Packt Publishing (2012)

[3] Adalsteinsson, D., Sethian, J.A.: A Fast Level Set Method for Propagating Interfaces. Journal of Computational Physics **118**(2), 269–277 (1995)

[4] de Aguiar, E., Stoll, C., Theobalt, C., Ahmed, N., Seidel, H., Thrun, S.: Performance Capture from Sparse Multi-view Video. ACM Transactions on Graphics (TOG) **27**(3) (2008)

[5] Alexandre, L.A.: 3D Descriptors for Object and Category Recognition: a Comparative Evaluation. In: Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2012)

[6] Amanatides, J., Woo, A.: A Fast Voxel Traversal Algorithm for Ray Tracing. In: Eurographics, vol. 87 (1987)

[7] Amberg, B., Romdhani, S., Vetter, T.: Optimal Step Nonrigid ICP Algorithms for Surface Registration. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2007)

[8] Andersen, M.R., Jensen, T., Lisouski, P., Mortensen, A.K., Hansen, M.K., Gregersen, T., Ahrendt, P.: Kinect Depth Sensor Evaluation for Computer Vision Applications. Technical report ECE-TR-6, Department of Engineering - Electrical and Computer Engineering, Aarhus University (2012)

[9] Angelini, E., Jin, Y., Laine, A.: State of the Art of Level Set Methods in Segmentation and Registration of Medical Imaging Modalities. Handbook of Biomedical Image Analysis: Registration Models **III** (2005)

[10] Aulinas, J., Petillot, Y., Salvi, J., Lladó, X.: The SLAM Problem: A Survey. In: Conference on Artificial Intelligence Research and Development (2008)

[11] Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.: PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. ACM Transactions on Graphics (TOG) **28**(3), 24:1–24:11 (2009)

[12] Ben-Chen, M., Butscher, A., Solomon, J., Guibas, L.: On Discrete Killing Vector Fields and Patterns on Surfaces. Computer Graphics Forum (CGF) **29**(5) (2010)

[13] Bernardini, F., Rushmeier, H.: The 3D Model Acquisition Pipeline. Computer Graphics Forum **21**(2), 149–172 (2002)

[14] Bertalmío, M., Cheng, L.T., Osher, S., Sapiro, G.: Variational Problems and Partial Differential Equations on Implicit Surfaces. Journal of Computational Physics **174**(2), 759–780 (2001)

[15] Besl, P.J., McKay, N.D.: A Method for Registration of 3-D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) **14**(2), 239–256 (1992)

[16] Blender Project: Free and Open 3D Creation Software. `https://www.blender.org/`. Last accessed: March 9, 2016

[17] Bleyer, M., Rhemann, C., Rother, C.: PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In: British Machine Vision Conference (BMVC) (2011)

[18] Bo, L., Ren, X., Fox, D.: Depth Kernel Descriptors for Object Recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2011)

[19] Bogo, F., Black, M.J., Loper, M., Romero, J.: Detailed Full-Body Reconstructions of Moving People from Monocular RGB-D Sequences. In: IEEE International Conference on Computer Vision (ICCV) (2015)

[20] Bogo, F., Romero, J., Loper, M., Black, M.J.: FAUST: Dataset and Evaluation for 3D Mesh Registration. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)

[21] Bogo, F., Romero, J., Pons-Moll, G., Black, M.J.: Dynamic FAUST: Registering Human Bodies in Motion. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[22] Bookstein, F.L.: Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **11**(6), 567–585 (1989)

[23] Botsch, M., Sorkine, O.: On Linear Variational Surface Deformation Methods. IEEE Transactions on Visualization and Computer Graphics (TVCG) **14**(1), 213–230 (2008)

[24] Bronstein, A.M.: Spectral Descriptors for Deformable Shapes. Tech. rep., School of Electrical Engineering, Tel Aviv University (2011)

[25] Brown, B., Rusinkiewicz, S.: Non-Rigid Range-Scan Alignment Using Thin-Plate Splines. In: 2nd International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT) (2004)

[26] Brown, B., Rusinkiewicz, S.: Global Non-Rigid Alignment of 3-D Scans. ACM Transactions on Graphics (TOG) **26**(3) (2007)

[27] Brown, L.G.: A Survey of Image Registration Techniques. ACM Computing Surveys (CSUR) **24**(4), 325–376 (1992)

[28] Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High Accuracy Optical Flow Estimation Based on a Theory for Warping. In: European Conference on Computer Vision (ECCV) (2004)

[29] Bylow, E., Olsson, C., Kahl, F.: Robust Camera Tracking by Combining Color and Depth Measurements. In: International Conference on Pattern Recognition (ICPR) (2014)

[30] Bylow, E., Sturm, J., Kerl, C., Kahl, F., Cremers, D.: Real-Time Camera Tracking and 3D Reconstruction Using Signed Distance Functions. In: Robotics: Science and Systems Conference (RSS) (2013)

[31] Cagniart, C., Boyer, E., Ilic, S.: Probabilistic Deformable Surface Tracking from Multiple Videos. In: European Conference on Computer Vision (ECCV) (2010)

[32] Calder, J., Mansouri, A., Yezzi, A.: Image Sharpening via Sobolev Gradient Flows. SIAM Journal on Imaging Sciences **3**(4), 981–1014 (2010)

[33] Canelhas, D.: sdf_tracker - ROS Wiki. `http://wiki.ros.org/sdf_tracker`. Last accessed: March 9, 2016

[34] Canelhas, D.R.: Scene Representation, Registration and Object Detection in a Truncated Signed Distance Function Representation of 3D Space. Master's thesis, Department of Technology, Örebro University (2012)

[35] Canelhas, D.R., Stoyanov, T., Lilienthal, A.J.: SDF Tracker: A Parallel Algorithm for On-line Pose Estimation and Scene Reconstruction from Depth Images. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)

[36] Chen, J., Bautembach, D., Izadi, S.: Scalable Real-time Volumetric Surface Reconstruction. ACM Transactions on Graphics **32**(4) (2013)

[37] Chen, Y., Medioni, G.: Object Modeling by Registration of Multiple Range Images. In: Proceedings of the 1991 IEEE International Conference on Robotics and Automation (ICRA), pp. 2724–2729, vol. 3 (1991)

[38] Choi, S., Zhou, Q., Miller, S., Koltun, V.: A Large Dataset of Object Scans. arXiv:1602.02481 (2016)

[39] Choi, S., Zhou, Q.Y., Koltun, V.: Robust Reconstruction of Indoor Scenes. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[40] Claes, P., Vandermeulen, D., Van Gool, L., Suetens, P.: Robust and Accurate Partial Surface Registration based on Variational Implicit Surfaces for Automatic 3D Model Building. In: Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM) (2005)

[41] Clarenz, U., Rumpf, M., Telea, A.: Robust Feature Detection and Local Classification for Surfaces Based on Moment Analysis. IEEE Transactions on Visualization and Computer Graphics **10**(5), 516–524 (2004)

[42] CloudCompare: 3D Point Cloud and Mesh Processing Software. `http://www.danielgm.net/cc/`. Last accessed: March 9, 2016

[43] Cohen-Or, D., Solomovic, A., Levin, D.: Three-dimensional Distance Field Metamorphosis. ACM Transactions on Graphics (TOG) **17**(2), 116–141 (1998)

[44] Collet, A., Chuang, M., Sweeney, P., Gillett, D., Evseev, D., Calabrese, D., Hoppe, H., Kirk, A., Sullivan, S.: High-Quality Streamable Free-Viewpoint Video. ACM Transactions on Graphics (TOG) **34**(4) (2015)

[45] Curless, B., Levoy, M.: A Volumetric Method for Building Complex Models from Range Images. In: 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '96, pp. 303–312 (1996)

[46] Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration. ACM Transactions on Graphics (TOG) **36**(4) (2017)

[47] Davison, A.J., Murray, D.W.: Simultaneous Localization and Map-Building Using Active Vision. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **24**(7), 865–880 (2002)

[48] Davison, A.J., Reid, I., Molton, N., Stasse, O.: MonoSLAM: Real-Time Single Camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **29**(6), 1052–1067 (2007)

[49] Dellaert, F., Seitz, S.M., Thorpe, C.E., Thrun, S.: Structure from motion without correspondence. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2000)

[50] Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum-likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society (1977)

[51] Diebel, J.: Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors. Tech. rep., Stanford University (2006)

[52] Dimashova, M., Lysenkov, I., Rabaud, V., Eruhimov, V.: Tabletop Object Scanning with an RGB-D Sensor. In: Third Workshop on Semantic Perception, Mapping and Exploration (SPME) at the 2013 IEEE International Conference on Robotics and Automation (ICRA) (2013)

[53] Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H.F., Csorba, M.: A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. IEEE Transactions on Robotics and Automation **17**(3), 229–241 (2001)

[54] Dou, M., Davidson, P., Fanello, S.R., Khamis, S., Kowdle, A., Rhemann, C., Tankovich, V., Izadi, S.: Motion2Fusion: Real-time Volumetric Performance Capture. In: ACM Transactions on Graphics (TOG) (2017)

[55] Dou, M., Khamis, S., Degtyarev, Y., Davidson, P., Fanello, S.R., Kowdle, A., Escolano, S.O., Rhemann, C., Kim, D., Taylor, J., Kohli, P., Tankovich, V., Izadi, S.: Fusion4D: Real-time Performance Capture of Challenging Scenes. ACM Transactions on Graphics (TOG) **35**(4) (2016)

[56] Dou, M., Taylor, J., Fuchs, H., Fitzgibbon, A., Izadi, S.: 3D Scanning Deformable Objects with a Single RGBD Sensor. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[57] Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model Globally, Match Locally: Efficient and Robust 3D Object Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2010)

[58] Eade, E.: Lie groups for 2D and 3D Transformations. Tech. rep. (2013)

[59] Eggert, D.W., Lorusso, A., Fisher, R.B.: Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms. Machine Vision and Application **9**(5-6), 272–290 (1997)

[60] Elfes, A., Matthies, L.: Sensor Integration for Robot Navigation: Combining Sonar and Stereo Range Data in a Grid-Based Representataion. In: 26th IEEE Conference on Decision and Control, vol. 26, pp. 1802–1807 (1987)

[61] Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., Burgard, W.: An Evaluation of the RGB-D SLAM System. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2012)

[62] Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-Scale Direct Monocular SLAM. In: European Conference on Computer Vision (ECCV) (2014)

[63] Enright, D., Marschner, S., Fedkiw, R.: Animation and Rendering of Complex Water Surfaces. ACM Transactions on Graphics (TOG) **21**(3), 736–744 (2002)

[64] Fanello, S.R., Rhemann, C., Tankovich, V., Kowdle, A., Escolano, S.O., Kim, D., Izadi, S.: HyperDepth: Learning Depth from Structured Light without Matching. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

[65] Fanello, S.R., Valentin, J., Rhemann, C., Kowdle, A., Tankovich, V., Davidson, P., Izadi, S.: UltraStereo: Efficient Learning-Based Matching for Active Stereo Systems. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[66] Farrell, J.A.: Computation of the Quaternion from a Rotation Matrix. Tech. rep., University of California, Riverside (2008)

[67] Feldmar, J., Ayache, N.: Rigid, Affine and Locally Affine Registration of Free-Form Surfaces. International Journal of Computer Vision **18**(2), 99–119 (1996)

[68] Fioraio, N., Taylor, J., Fitzgibbon, A., Di Stefano, L., Izadi, S.: Large-Scale and Drift-Free Surface Reconstruction Using Online Subvolume Registration. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[69] Fischler, M.A., Bolles, R.C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM **24**(6), 381–395 (1981)

[70] Fitzgibbon, A., Zisserman, A.: Automatic Camera Recovery for Closed or Open Image Sequences. In: 5th European Conference on Computer Vision (ECCV) (1998)

[71] Fitzgibbon, A.W.: Robust Registration of 2D and 3D Point Sets. In: Proceedings of the British Machine Vision Conference (BMVC), pp. 1–10 (2001)

[72] Frank, A.: On Kuhnâ€™s Hungarian Method - A Tribute from Hungary. Tech. Rep. 2004-14, Egérvary Research Group on Combinatorial Optimization, Budapest, Hungary (2004)

[73] Frisken, S.F., Perry, R.N.: Designing with Distance Fields. In: ACM SIGGRAPH 2006 Courses, SIGGRAPH ’06, pp. 60–66 (2006)

[74] Fujiwara, K., Nishino, K., Takamatsu, J., Zheng, B., Ikeuchi, K.: Locally Rigid Globally Non-rigid Surface Registration. In: IEEE International Conference on Computer Vision (ICCV) (2011)

[75] Gall, J., Stoll, C., de Aguiar, E., Theobalt, C., Rosenhahn, B., Seidel, H.P.: Motion Capture Using Joint Skeleton Tracking and Surface Estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2009)

[76] Gelfand, N., Mitra, N.J., Guibas, L.J., Pottmann, H.: Robust Global Registration. In: Proceedings of the Third Eurographics Symposium on Geometry Processing (SGP) (2005)

[77] Goesele, M., Curless, B., Seitz, S.M.: Multi-View Stereo Revisited. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (2006)

[78] Gokturk, S.B., Yalcin, H., Bamji, C.: A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2004)

[79] Gonzalez, F.B.: Lie Algebras. Lecture notes, Tufts University (2007)

[80] Guo, K., Xu, F., Wang, Y., Liu, Y., Dai, Q.: Robust Non-rigid Motion Tracking and Surface Reconstruction Using L0 Regularization. In: IEEE International Conference on Computer Vision (ICCV) (2015)

[81] Guo, K., Xu, F., Yu, T., Liu, X., Dai, Q., Liu, Y.: Real-time Geometry, Albedo and Motion Reconstruction Using a Single RGBD Camera. ACM Transactions on Graphics (TOG) (2017)

[82] Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM. In: IEEE International Conference on Robotics and Automation (ICRA) (2014)

[83] Hartley, R., Sturm, P.: Triangulation. Computer Vision and Image Understanding (CVIU) **68**(2), 146–157 (1997)

[84] Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, New York, USA (2003)

[85] Henry, P., Fox, D., Bhowmik, A., Mongia, R.: Patch Volumes: Segmentation-Based Consistent Mapping with RGB-D Cameras. In: International Conference on 3D Vision (3DV) (2013)

[86] Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. In: International Symposium on Experimental Robotics (2010)

[87] Ho, S., Bullitt, E., Gerig, G.: Level-Set Evolution with Region Competition: Automatic 3-D Segmentation of Brain Tumors. In: 16th International Conference on Pattern Recognition (ICPR) (2002)

[88] Hodaň, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., Zabulis, X.: T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects (2017)

[89] Holzer, S., Shotton, J., Kohli, P.: Learning to Efficiently Detect Repeatable Interest Points in Depth Data. In: 12th European Conference on Computer Vision (ECCV) (2012)

[90] Horn, B., Brooks, M.: Shape from Shading. MIT Press (1989)

[91] Horn, B.K.P.: Closed-Form Solution of Absolute Orientation Using Unit Quaternions. Journal of the Optical Society of America A **4**(4), 629–642 (1987)

[92] Hornung, A., Wurm, K., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: an Efficient Probabilistic 3D Mapping Framework Based on Octrees. Autonomous Robots **34**(3), 189–206 (2013)

[93] Houston, B., Nielsen, M.B., Batty, C., Nilsson, O., Museth, K.: Hierarchical RLE Level Set: A Compact and Versatile Deformable Surface Representation. ACM Transactions on Graphics (TOG) **25**(1), 151–175 (2006)

[94] Huang, C., Cagniart, C., Boyer, E., Ilic, S.: A Bayesian Approach to Multiview 4D Modeling. International Journal of Computer Vision (IJCV) **116**(2), 115–135 (2016)

[95] Huang, C.H., Allain, B., Franco, J.S., Navab, N., Ilic, S., Boyer, E.: Volumetric 3D Tracking by Detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

[96] Huber, D.F., Hebert, M.: Fully Automatic Registration of Multiple 3D Data Sets. Image and Vision Computing **21**(7), 637–650 (2003)

[97] Huguet, F., Devernay, F.: A Variational Method for Scene Flow Estimation from Stereo Sequences. In: IEEE International Conference on Computer Vision (ICCV) (2007)

[98] Innmann, M., Zollhöfer, M., Nießner, M., Theobalt, C., Stamminger, M.: VolumeDeform: Real-time Volumetric Non-rigid Reconstruction. In: European Conference on Computer Vision (ECCV) (2016)

[99] Ioannou, Y., Taati, B., Harrap, R., Greenspan, M.A.: Difference of Normals as a Multi-scale Operator in Unorganized Point Clouds. In: Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission (3DIMPVT) (2012)

[100] Irani, M., Anandan, P.: About Direct Methods. In: Vision Algorithms: Theory and Practice (2000)

[101] Ivancevic, V.G., Ivancevic, T.T.: Lecture Notes in Lie Groups. Tech. rep., ArXiV e-prints (2011)

[102] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In: ACM Symposium on User Interface Software and Technology (UIST) (2011)

[103] Jaimez, M., Kerl, C., Gonzalez-Jimenez, J., Cremers, D.: Fast Odometry and Scene Flow from RGB-D Cameras Based on Geometric Clustering. In: IEEE International Conference on Robotics and Automation (ICRA) (2017)

[104] Johnson, A., Kang, S.B.: Registration and Integration of Textured 3D Data. Image and Vision Computing **17**

[105] Johnson, A.E., Hebert, M.: Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) **21**(5), 433–449 (1999)

[106] Jones, M., Baerentzen, J.A., Sramek, M.: 3D Distance Fields: A Survey of Techniques and Applications. IEEE Transactions on Visualization and Computer Graphics **12**(4), 581–599 (2006)

[107] Joo, H., Liu, H., Tan, L., Gui, L., Nabbe, B., Matthews, I., Kanade, T., Nobuhara, S., Sheikh, Y.: Panoptic Studio: A Massively Multiview System for Social Motion Capture. In: IEEE International Conference on Computer Vision (ICCV) (2015)

[108] Kähler, O., Prisacariu, V., Valentin, J., Murray, D.: Hierarchical Voxel Block Hashing for Efficient Integration of Depth Images. IEEE Robotics and Automation Letters **1**(1), 192–197 (2016)

[109] Kähler, O., Prisacariu, V.A., Ren, C.Y., Sun, X., Torr, P., Murray, D.: Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices. IEEE Transactions on Visualization and Computer Graphics (TVCG) **21**(11), 1241–1250 (2015)

[110] Kähler, O., Prisacariu, V.A., Ren, C.Y., Sun, X., Torr, P., Murray, D.: Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices. IEEE Transactions on Visualization and Computer Graphics (TVCG) **21**(11), 1241–1250 (2015)

[111] Kehl, W., Navab, N., Ilic, S.: Coloured Signed Distance Fields for full 3D Object Reconstruction. In: Proceedings of the British Machine Vision Conference (BMVC) (2014)

[112] Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., Kolb, A.: Realtime 3D Reconstruction in Dynamic Scenes using Point-based Fusion. In: 2013 International Conference on 3D Vision (3DV) (2013)

[113] Kerl, C.: GitHub - tum-vision/dvo: Dense Visual Odometry. `https://github.com/tum-vision/dvo`. Last accessed: March 9, 2016

[114] Kerl, C., Sturm, J., Cremers, D.: Dense Visual SLAM for RGB-D Cameras. In: International Conference on Intelligent Robot Systems (IROS) (2013)

[115] Kerl, C., Sturm, J., Cremers, D.: Robust Odometry Estimation for RGB-D Cameras. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)

[116] Khoshelham, K., Elberink, S.O.: Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. Sensors **12**(2), 1437–1454 (2012)

[117] Klein, G., Murray, D.: Parallel Tracking and Mapping for Small AR Workspaces. In: Proceedings of the Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR) (2007)

[118] Kozlov, C., Slavcheva, M., Ilic, S.: Patch-based Non-rigid 3D Reconstruction from a Single Depth Stream. In: International Conference on 3D Vision (3DV) (2018)

[119] Kruskal, J.B.: Multiway Data Analysis. chap. Rank, Decomposition, and Uniqueness for 3-way and N-way Arrays (1989)

[120] Kubacki, D.B.: Signed Distance Registration for Depth Image Sequence. Master's thesis, University of Illinois at Urbana-Champaign (2011)

[121] Kubacki, D.B., Bui, H.Q., Babacan, S.D., Do, M.N.: Registration and Integration of Multiple Depth Images using Signed Distance Function. In: SPIE Proceedings, vol. 8296 (2012)

[122] Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g2o: A General Framework for Graph Optimization. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3607–3613 (2011)

[123] Lai, K., Bo, L., Ren, X., Fox, D.: A Large-Scale Hierarchical Multi-View RGB-D Object Dataset. In: IEEE International Conference on Robotics and Automation (ICRA) (2011)

[124] Laurentini, A.: The Visual Hull Concept for Silhouette-Based Image Understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **16**(2), 150–162 (1994)

[125] Lee, T., Lai, S.: 3D Non-rigid Registration for MPU Implicit Surfaces. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2008)

[126] Leonard, J.J., Durrant-Whyte, H.F.: Simultaneous Map Building and Localization for an Autonomous Mobile Robot. In: IEEE/RSJ Intelligent Robots and Systems International Workshop on Intelligence for Mechanical Systems (IROSW) (1991)

[127] Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., Fulk, D.: The Digital Michelangelo Project: 3D Scanning of Large Statues. In: Proceedings of ACM SIGGRAPH 2000, pp. 131–144 (2000)

[128] Levy, B.: Laplace-Beltrami Eigenfunctions Towards an Algorithm That "Understands" Geometry. In: IEEE International Conference on Shape Modeling and Applications (SMI) (2006)

[129] Li, C., Xu, C., Gui, C., Fox, M.D.: Level Set Evolution Without Re-initialization: A New Variational Formulation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2005)

[130] Li, C., Xu, C., Gui, C., Fox, M.D.: Distance Regularized Level Set Evolution and Its Application to Image Segmentation. IEEE Transaction on Image Processing (TIP) **19**(12), 3243–3254 (2010)

[131] Li, H., Adams, B., Guibas, L., Pauly, M.: Robust Single-View Geometry and Motion Reconstruction. ACM Transactions on Graphics (TOG) **28**(5) (2009)

[132] Li, H., Sumner, R., Pauly, M.: Global Correspondence Optimization for Non-Rigid Registration of Depth Scans. Computer Graphics Forum (CGF) **27**(5), 1421–1430

[133] Litany, O., Rodolà, E., Bronstein, A.M., Bronstein, M.M.: Fully Spectral Partial Shape Matching. Computer Graphics Forum (CGF) **36**(2), 247–258 (2017)

[134] Litany, O., Rodolà, E., Bronstein, A.M., Bronstein, M.M., Cremers, D.: Non-Rigid Puzzles. Computer Graphics Forum (CGF) **35**(5), 135–143 (2016)

[135] Litman, R., Bronstein, A.M., Bronstein, M.M.: Stable Volumetric Features in Deformable Shapes. Computers & Graphics **36**(5), 569–576 (2012)

[136] Lorensen, W.E., Cline, H.E.: Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '87, pp. 163–169 (1987)

[137] Losasso, F., Fedkiw, R., Osher, S.: Spatially Adaptive Techniques for Level Set Methods and Incompressible Flow. Computers and Fluids **35**(10), 995–1010 (2006)

[138] Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision (IJCV) **60**(2), 91–110 (2004)

[139] Lucas, B.C., Kazhdan, M., Taylor, R.H.: SpringLS: A Deformable Model Representation to Provide Interoperability between Meshes and Level Sets. In: 14th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI) (2011)

[140] Lucas, B.C., Kazhdan, M., Taylor, R.H.: Spring Level Sets: A Deformable Model Representation to Provide Interoperability between Meshes and Level Sets. IEEE Transactions on Visualization and Computer Graphics (VCG) **19**(5), 852–865 (2013)

[141] Ma, Y., Soatto, S., Kosecka, J., Sastry, S.S.: An Invitation to 3-D Vision: From Images to Geometric Models. Springer Verlag (2003)

[142] Machline, M., Arieli, Y., Sphpunt, A., Freedman, B.: Depth Mapping Using Projected Patterns (2010). Prime Sense Ltd., US patent 20100118123

[143] Magnor, M.A., Grau, O., Sorkine-Hornung, O., Theobalt, C.: Digital Representations of the Real World: How to Capture, Model, and Render Visual Reality. CRC Press (2015)

[144] Maintz, J., Viergever, M.: A Survey of Medical Image Registration. Medical Image Analysis **2**(1), 1–36 (1998)

[145] Makadia, A., Patterson, A., Daniilidis, K.: Fully Automatic Registration of 3D Point Clouds. In: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1297–1304 (2006)

[146] Masuda, T.: Registration and Integration of Multiple Range Images by Matching Signed Distance Fields for Object Shape Modeling. Computer Vision and Image Understanding (CVIU) **87**(1-3), 51–65 (2002)

[147] Mateus, D., Horaud, R., Knossow, D., Cuzzolin, F., Boyer, E.: Articulated Shape Matching using Laplacian Eigenfunctions and Unsupervised Point Registration. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2008)

[148] Meilland, M., Comport, A.I.: On Unifying Key-frame and Voxel-based Dense Visual SLAM at Large Scales. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)

[149] Mihalef, V.: The Marker Level Set Method: Applications to Simulation of Liquids. Ph.D. thesis (2007)

[150] Mihalef, V., Metaxas, D., Sussman, M.: Textured Liquids based on the Marker Level Set. Computer Graphics Forum (CGF) **26**(3), 457–466 (2007)

[151] Möbius, A.F.: Der Barycentrische Calcul (1827)

[152] Moravec, H.P.: Robot Spatial Perception by Stereoscopic Vision and 3D Evidence Grids. Tech. rep., The Robotic Institute, Carnegie Mellon University (1996)

[153] Neubeck, A., Van Gool, L.: Efficient Non-Maximum Suppression. In: 18th International Conference on Pattern Recognition (ICPR) (2006)

[154] Neuberger, J.: Sobolev Gradients and Differential Equations. Springer Science & Business Media (2009)

[155] Neugebauer, P.J.: Geometrical Cloning of 3D Objects via Simultaneous Registration of Multiple Range Images. In: Proceedings of the International Conference on Shape Modeling and Applications, pp. 130–139 (1997)

[156] Newcombe, R.A., Fox, D., Seitz, S.M.: DynamicFusion: Reconstruction and Tracking of Non-rigid Scenes in Real-Time. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[157] Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: Real-Time Dense Surface Mapping and Tracking. In: 10th International Symposium on Mixed and Augmented Reality (ISMAR) (2011)

[158] Newcombe, R.A., Lovegrove, S.J., Davison, A.J.: DTAM: Dense tracking and mapping in real-time. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2320–2327 (2011)

[159] Nguyen, C.V., Izadi, S., Lovell, D.: Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking. In: Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission (3DIMPVT) (2012)

[160] Nielsen, M.B., Museth, K.: Dynamic Tubular Grid: An Efficient Data Structure and Algorithms for High Resolution Level Sets. Journal of Scientific Computing **26**(3), 261–299 (2006)

[161] Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3D Reconstruction at Scale using Voxel Hashing. ACM Transactions on Graphics (TOG) (2013)

[162] Nister, D., Naroditsky, O., Bergen, J.: Visual Odometry. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (2004)

[163] Osher, S., Fedkiw, R.: Level Set Methods and Dynamic Implicit Surfaces, *Applied Mathematical Science*, vol. 153. Springer (2003)

[164] Osher, S., Sethian, J.: Fronts Propagating with Curvature-dependent speed: Algorithms based on Hamilton-Jacobi Formulations. Journal of Computational Physics **79**(1), 12–49 (1988)

[165] Paragios, N., Rousson, M., Ramesh, V.: Non-Rigid Registration Using Distance Functions. Computer Vision and Image Understanding **89**(2-3), 142–165 (2003)

[166] Parker, S., Shirley, P., Livnat, Y., Hansen, C., Sloan, P.P.: Interactive Ray Tracing for Isosurface Rendering. In: IEEE Visualization (1998)

[167] PCL: Point Cloud Library. `http://pointclouds.org/`. Last accessed: March 9, 2016

[168] Pirker, K., Rüther, M., Schweighofer, G., Bischof, H.: GPSlam: Marrying Sparse Geometric and Dense Probabilistic Visual Mapping. In: Proceedings of the British Machine Vision Conference (BMVC) (2011)

[169] Pons, J.P., Hermosillo, G., Keriven, R., Faugeras, O.: How to Deal with Point Correspondences and Tangential Velocities in the Level Set Framework. In: IEEE International Conference on Computer Vision (ICCV) (2003)

[170] Pulli, K.: Multiview registration for large data sets. In: Proceedings of the Second International Conference on 3-D Digital Imaging and Modeling, pp. 160–168 (1999)

[171] Quiroga, J., Brox, T., Devernay, F., Crowley, J.: Dense Semi-rigid Scene Flow Estimation from RGBD Images. In: European Conference on Computer Vision (ECCV) (2014)

[172] Raviv, D., Bronstein, M.M., Bronstein, A.M., Kimmel, R.: Volumetric Heat Kernel Signatures. In: ACM Workshop on 3D Object Retrieval (2010)

[173] Reuter, M., Wolter, F.E., Peinecke, N.: Laplace-spectra as Fingerprints for Shape Matching. In: ACM Symposium on Solid and Physical Modeling (2005)

[174] Reuter, M., Wolter, F.E., Peinecke, N.: Laplace-Beltrami Spectra As 'Shape-DNA' of Surfaces and Solids. Computer-Aided Design **38**(4), 342–366 (2006)

[175] Rodolà, E., Cosmo, L., Bronstein, M.M., Torsello, A., Cremers, D.: Partial Functional Correspondence. Computer Graphics Forum (CGF) **36**(1), 222–236 (2017)

[176] Roth, H., Vona, M.: Moving Volume KinectFusion. In: British Machine Vision Conference (BMVC) (2012)

[177] Rouhani, M., Sappa, A.D.: The Richer Representation the Better Registration. IEEE Transactions on Image Processing **22**(12), 5036–5049 (2013)

[178] Ruhnke, M., Kümmerle, R., Grisetti, G., Burgard, W.: Range Sensor Based Model Construction by Sparse Surface Adjustment. In: IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO), pp. 46–49 (2011)

[179] Rünz, M., Agapito, L.: Co-Fusion: Real-time Segmentation, Tracking and Fusion of Multiple Objects. In: IEEE International Conference on Robotics and Automation (ICRA) (2017)

[180] Rusinkiewicz, S., Levoy, M.: Efficient Variants of the ICP Algorithm. In: 3rd International Conference on 3D Digital Imaging and Modeling (3DIM) (2001)

[181] Rustamov, R.M.: Laplace-Beltrami Eigenfunctions for Deformation Invariant Shape Representation. In: Eurographics Symposium on Geometry Processing (SGP) (2007)

[182] Rustamov, R.M.: Interpolated Eigenfunctions for Volumetric Shape Processing. The Visual Computer **27**(11) (2011)

[183] Rusu, R.B., Holzbach, A., Blodow, N., Beetz, M.: Fast Geometric Point Labeling using Conditional Random Fields. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2009)

[184] Salvi, J., Pagès, J., Batlle, J.: Pattern Codification Strategies in Structured Light Systems. Pattern Recognition **37**, 827–849 (2004)

[185] Schmidt, T., Newcombe, R., Fox, D.: Self-Supervised Visual Descriptor Learning for Dense Correspondence. IEEE Robotics and Automation Letters **2**(2), 420–427 (2017)

[186] Schütz, C., Jost, T., Hugli, H.: Multi-Feature Matching Algorithm for Free-Form 3D Surface Registration. In: Proceedings of the Fourteenth International Conference on Pattern Recognition (ICPR), pp. 982–984, vol. 2 (1998)

[187] Sederberg, T., Parry, S.: Free-form Deformation of Solid Geometric Models. In: 13th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH) (1986)

[188] Segal, A., Haehnel, D., Thrun, S.: Generalized-ICP. In: Proceesings of Robotics: Science and Systems (RSS) (2009)

[189] Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (2006)

[190] Sethian, J.: Level Set Methods and Fast Marching Methods. Cambridge University Press (1999)

[191] Shih, S.W., Chuang, Y.T., Yu, T.Y.: An Efficient and Accurate Method for the Relaxation of Multiview Registration Error. IEEE Transactions on Image Processing **17**(6), 968–981 (2008)

[192] Singh, A., Sha, J., Narayan, K., Achim, T., Abbeel, P.: BigBIRD: A Large-Scale 3D Database of Object Instances. In: IEEE International Conference on Robotics and Automation (ICRA) (2014)

[193] Slavcheva, M., Baust, M., Cremers, D., Ilic, S.: KillingFusion: Non-rigid 3D Reconstruction without Correspondences. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[194] Slavcheva, M., Baust, M., Ilic, S.: Towards Implicit Correspondence in Signed Distance Field Evolution. In: PeopleCap Workshop, IEEE International Conference on Computer Vision (ICCVW) (2017)

[195] Slavcheva, M., Baust, M., Ilic, S.: SobolevFusion: 3D Reconstruction of Scenes Undergoing Free Non-rigid Motion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[196] Slavcheva, M., Baust, M., Ilic, S.: Variational Level Set Evolution for Non-rigid 3D Reconstruction from a Single Depth Camera. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) (2018)

[197] Slavcheva, M., Ilic, S.: SDF-TAR: Parallel Tracking and Refinement in RGB-D Data using Volumetric Registration. In: British Machine Vision Conference (BMVC) (2016)

[198] Slavcheva, M., Kehl, W., Navab, N., Ilic, S.: SDF-2-SDF: Highly Accurate 3D Object Reconstruction. In: European Conference on Computer Vision (ECCV) (2016)

[199] Slavcheva, M., Kehl, W., Navab, N., Ilic, S.: SDF-2-SDF Registration for Real-time 3D Reconstruction from RGB-D Data. International Journal of Computer Vision (IJCV) **126**(6), 615–636 (2017)

[200] Solomon, J., Ben-Chen, M., Butscher, A., Guibas, L.: As-Killing-As-Possible Vector Fields for Planar Deformation. Computer Graphics Forum (CGF) **30**(5) (2011)

[201] Sorkine, O., Alexa, M.: As-Rigid-As-Possible Surface Modeling. In: Fifth Eurographics Symposium on Geometry Processing (SGP) (2007)

[202] Steder, B., Rusu, R.B., Konolige, K., Burgard, W.: NARF: 3D Range Image Features for Object Recognition. In: Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2010)

[203] Steinbrücker, F., Kerl, C., Sturm, J., Cremers, D.: Large-Scale Multi-Resolution Surface Reconstruction from RGB-D Sequences. In: IEEE International Conference on Computer Vision (ICCV) (2013)

[204] Steinbrücker, F., Sturm, J., Cremers, D.: Real-Time Visual Odometry from Dense RGB-D Images. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops) (2011)

[205] Steinbrücker, F., Sturm, J., Cremers, D.: Volumetric 3D Mapping in Real-time on a CPU. In: IEEE International Conference on Robotics and Automation (ICRA) (2014)

[206] Stoyanov, T., Magnusson, M., Lilienthal, A.: Point Set Registration through Minimization of the L_2 Distance between 3D-NDT Models. In: IEEE International Conference on Robotics and Automation (ICRA) (2012)

[207] Stühmer, J., Gumhold, S., Cremers, D.: Real-time Dense Geometry from a Handheld Camera. In: 32nd DAGM Symposium on Pattern Recognition (2010)

[208] Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A Benchmark for the Evaluation of RGB-D SLAM Systems. In: Proceedings of the International Conference on Intelligent Robot Systems (IROS) (2012)

[209] Sumner, R.W., Schmid, J., Pauly, M.: Embedded Deformation for Shape Manipulation. ACM Transactions on Graphics (TOG) **26**(3) (2007)

[210] Sun, J., Ovsjanikov, M., Guibas, L.: A Concise and Provably Informative Multi-scale Signature Based on Heat Diffusion. In: Proceedings of the Symposium on Geometry Processing (SGP) (2009)

[211] Sundaramoorthi, G., Yezzi, A., Mennucci, A.: Coarse-to-Fine Segmentation and Tracking using Sobolev Active Contours. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **30**(5), 851–864 (2008)

[212] Sundaramoorthi, G., Yezzi, A., Mennucci, A.C.: Sobolev Active Contours. International Journal of Computer Vision (IJCV) **73**(3), 345–366 (2007)

[213] Tamaki, T., Abe, M., Raytchev, B., Kaneda, K.: Softassign and EM-ICP on GPU. In: First International Conference on Networking and Computing (2010)

[214] Tan, D.J., Cashman, T., Taylor, J., Fitzgibbon, A., Tarlow, D., Khamis, S., Izadi, S., Shotton, J.: Fits Like a Glove: Rapid and Reliable Hand Shape Personalization. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

[215] Tao, M., Solomon, J., Butscher, A.: Near-Isometric Level Set Tracking. Computer Graphics Forum (CGF) **35**(5) (2016)

[216] Taylor, J., Bordeaux, L., Cashman, T., Corish, B., Keskin, C., Sharp, T., Soto, E., Sweeney, D., Valentin, J., Luff, B., Topalian, A., Wood, E., Khamis, S., Kohli, P., Izadi, S., Banks, R., Fitzgibbon, A., Shotton, J.: Efficient and Precise Interactive Hand Tracking Through Joint, Continuous Optimization of Pose and Correspondences. ACM Transactions on Graphics (TOG) **35**(4) (2016)

[217] Terzopoulos, D., Platt, J., Barr, A., Fleischer, K.: Elastically Deformable Models. Computer Graphics **21**(4), 205–214 (1987)

[218] Thies, J., Zollhöfer, M., Nießner, M., Valgaerts, L., Stamminger, M., Theobalt, C.: Real-time Expression Transfer for Facial Reenactment. ACM Transactions on Graphics (TOG) **34**(6) (2015)

[219] Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., Nießner, M.: Face2Face: Real-time Face Capture and Reenactment of RGB Videos. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

[220] Tomasi, C., Manduchi, R.: Bilateral Filtering for Gray and Color Images. In: Sixth IEEE International Conference on Computer Vision (ICCV), pp. 839–846 (1998)

[221] Tombari, F., Salti, S., Di Stefano, L.: Performance Evaluation of 3D Keypoint Detectors. International Journal of Computer Vision (IJCV) **102**(1), 198–220 (2013)

[222] Tsai, R.: A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology using Off-the-shelf TV Cameras and Lenses. IEEE Journal on Robotics and Automation **3**(4), 323–344 (1987)

[223] Turk, G., O'Brien, J.: Shape Transformation Using Variational Implicit Functions. In: 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99 (1999)

[224] Tykkälä, T., Audras, C., Comport, A.: Direct Iterative Closest Point for real-time visual odometry. In: IEEE International Conference on Computer Vision Workshops (ICCVW) (2011)

[225] Umeyama, S.: An Eigendecomposition Approach to Weighted Graph Matching Problems. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **10**(5), 695–703 (1988)

[226] Van Verth, J.: Understanding Rotations. Tech. rep., Game Developers Conference (2012)

[227] Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-Dimensional Scene Flow. In: IEEE International Conference on Computer Vision (ICCV) (1999)

[228] Vijayanagar, K.R., Loghman, M., Kim, J.: Real-Time Refinement of Kinect Depth Maps using Multi-Resolution Anisotropic Diffusion. Mobile Networks and Applications **19**(3), 414–425 (2014)

[229] Vlasic, D., Baran, I., Matusik, W., Popović, J.: Articulated Mesh Animation from Multi-view Silhouettes. ACM Transactions on Graphics (TOG) **27**(3) (2008)

[230] Vlasic, D., Peers, P., Baran, I., Debevec, P., Popović, J., Rusinkiewicz, S., Matusik, W.: Dynamic Shape Capture Using Multi-view Photometric Stereo. ACM Transactions on Graphics (TOG) **28**(5) (2009)

[231] Vogel, C., Schindler, K., Roth, S.: Piecewise Rigid Scene Flow. In: IEEE International Conference on Computer Vision (ICCV) (2013)

[232] Wasenmüller, O., Ansari, M., Stricker, D.: DNA-SLAM: Dense Noise Aware SLAM for ToF RGB-D Cameras. In: Asian Conference on Computer Vision (ACCV) International Workshops (2016)

[233] Wasenmüller, O., Meyer, M., Stricker, D.: CoRBS: Comprehensive RGB-D Benchmark for SLAM using Kinect v2. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2016)

[234] Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., Cremers, D.: Efficient Dense Scene Flow from Sparse or Dense Stereo Data. In: 10th European Conference on Computer Vision (ECCV) (2008)

[235] Weiszfeld, E., Plastria, F.: On the Point for Which the Sum of the Distances to n Given Points is Minimum. Tôhoku Mathematical Journal (1937)

[236] Weng, Y., Chai, M., Xu, W., Tong, Y., Zhou, K.: As-Rigid-As-Possible Distance Field Metamorphosis. Computer Graphics Forum (CGF) **32**(7), 381–389 (2013)

[237] Whelan, T., Johannsson, H., Kaess, M., Leonard, J.J., McDonald, J.B.: Robust Real-Time Visual Odometry for Dense RGB-D Mapping. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)

[238] Whelan, T., Kaess, M., Leonard, J.J., McDonald, J.: Deformation-based loop closure for large scale dense rgb-d slam. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)

[239] Whelan, T., Leutenegger, S., Salas-Moreno, R.F., Glocker, B., Davison, A.J.: ElasticFusion: Dense SLAM Without A Pose Graph. In: Robotics: Science and Systems (RSS) (2015)

[240] Whelan, T., McDonald, J.B., Kaess, M., Fallon, M.F., Johannsson, H., Leonard, J.J.: Kintinuous: Spatially Extended KinectFusion. In: RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras (2012)

[241] Whelan, T., Salas-Moreno, R.F., Glocker, B., Davison, A.J., Leutenegger, S.: ElasticFusion: Real-Time Dense SLAM and Light Source Estimation. The International Journal of Robotics Research (IJRR) **35**(14), 1697–1716 (2016)

[242] Whitaker, R.T.: A Level-Set Approach to 3D Reconstruction from Range Data. International Journal of Computer Vision (IJCV) **29**(3), 203–231 (1998)

[243] Williams, O., Fitzgibbon, A.: Gaussian process implicit surfaces. Gaussian Processes in Practice (2007)

[244] Woodham, R.J.: Photometric Method for Determining Surface Orientation from Multiple Images. Optical Engineering **19**(I), 139–144 (1980)

[245] Wulff, J., Black, M.J.: Efficient Sparse-to-Dense Optical Flow Estimation using a Learned Basis and Layers. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[246] Ye, Q.Z.: The Signed Euclidean Distance Transform and Its Applications. In: 9th International Conference on Pattern Recognition, vol. 1, pp. 495–499 (1988)

[247] Yu, T., Guo, K., Xu, F., Dong, Y., Su, Z., Zhao, J., Li, J., Dai, Q., Liu, Y.: BodyFusion: Real-time Capture of Human Motion and Surface Geometry Using a Single Depth Camera. In: IEEE International Conference on Computer Vision (ICCV) (2017)

[248] Yu, T., Zheng, Z., Guo, K., Zhao, J., Dai, Q., Li, H., Pons-Moll, G., Liu, Y.: DoubleFusion: Real-time Capture of Human Performances with Inner Body Shapes from a Single Depth Sensor. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[249] Zach, C., Pock, T., Bischof, H.: A Globally Optimal Algorithm for Robust TV-L^1 Range Image Integration. In: Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV), pp. 1–8 (2007)

[250] Zaharescu, A., Boyer, E., Horaud, R.: Topology-Adaptive Mesh Deformation for Surface Evolution, Morphing, and Multiview Reconstruction. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **33**(4), 823–837 (2011)

[251] ZCorporation: ZPrinter 650. Hardware manual (2008)

[252] Zeng, M., Zhao, F., Zheng, J., Liu, X.: A Memory-efficient Kinectfusion Using Octree. In: Proceedings of the First International Conference on Computational Visual Media, CVM'12 (2012)

[253] Zeng, M., Zhao, F., Zheng, J., Liu, X.: Octree-based Fusion for Realtime 3D Reconstruction. Graphical Models **75**(3), 126–136 (2013)

[254] Zhang, Z.: Iterative Point Matching for Registration of Free-Form Curves and Surfaces. International Journal of Computer Vision **13**(2), 119–152 (1994)

[255] Zhang, Z.: A Flexible New Technique for Camera Calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **22**(11), 1330–1334 (2000)

[256] Zhao, H.K., Chan, T., Merriman, B., Osher, S.: A Variational Level Set Approach to Multiphase Motion. Journal of Computational Physics **127**(1), 179–195 (1996)

[257] Zhou, Q., Koltun, V.: Dense Scene Reconstruction with Points of Interest. ACM Transactions on Graphics **32**(4) (2013)

[258] Zhou, Q., Miller S. Koltun, V.: Elastic Fragments for Dense Scene Reconstruction. In: IEEE International Conference on Computer Vision (ICCV) (2013)

[259] Zhou, Q.Y., Park, J., Koltun, V.: Fast Global Registration. In: European Conference on Computer Vision (ECCV) (2016)

[260] Zikic, D., Kamen, A., Navab, N.: Natural Gradients for Deformable Registration. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2010)

[261] Zollhöfer, M., Nießner, M., Izadi, S., Rhemann, C., Zach, C., Fisher, M., Wu, C., Fitzgibbon, A., Loop, C., Theobalt, C., Stamminger, M.: Real-time Non-rigid Reconstruction using an RGB-D Camera. ACM Transactions on Graphics (TOG) **33**(4) (2014)

[262] Zollhöfer, M., Sert, E., Greiner, G., Süßmuth, J.: GPU based ARAP Deformation using Volumetric Lattices. In: Eurographics - Short Papers (2012)

[263] Zollhöfer, M., Stotko, P., Görlitz, A., Theobalt, C., Nießner, M., Klein, R., Kolb, A.: State of the Art on 3D Reconstruction with RGB-D Cameras. In: Eurographics - State-of-the-Art Reports (STARs), vol. 37 (2018)